A CALCULUS OF VARIATIONS METHOD FOR PLANNING PATHS AROUND OBSTACLES IN EUCLIDEAN SPACE

By

JOHN DAMIAN DUFFY

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1994

ACKNOWLEDGEMENTS

I would like to thank Dr. A. Seireg for serving as my advisor during the period of study for this research. His invaluable assistance and skill in guiding me have helped make this experience very productive and enjoyable. I would like to thank the faculty, staff and students of CIMAR and the Department of Mechanical Engineering who have helped me over the past three years. In particular I would like to thank Dr. Jose Maria Rico Martinez, a visiting scholar who provided ideas and criticism in the early stages work, and Dr. Philip Adsit for helping me with the software and hardware to prepare this manuscript.

My extended family has provided me with constant encouragement during the past three years. I would especially like to thank my parents Anne and Joseph Duffy for their advice and guidance during my university career. Most importantly, I would like to thank my wife Nadia for her love and understanding during the many hours I have had to spend working on this dissertation. Without her support and giving me the freedom to devote my time to this work, it would not have been possible.

I would also like to thank CIMAR and the Department of Mechanical Engineering for the financial support and the use of equipment and facilities which have made this research possible.

TABLE OF CONTENTS

Abstract of Dissertation Presented to the Graduate School of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

A CALCULUS OF VARIATIONS METHOD FOR PLANNING PATHS AROUND OBSTACLES IN EUCLIDEAN SPACE

By

JOHN DAMIAN DUFFY

December 1994

Chairman: Dr. Ali Seireg
Major Department: Mechanical Engineering

This dissertation presents a new method for path planning around obstacles in Euclidean space using variational calculus. The method is derived from differential geometry and Riemannian manifolds. In particular the problem formulation calls for minimizing the energy function over the Riemannian manifold subject to the obstacle constraints. For the path planning problem it is shown that there is an equivalence between minimizing the energy function for a given curve and its length. From a variational calculus point of view, the path planning formulation falls under the classification of the Problem of Bolza. The constrained problem is transformed into an unconstrained minimization in a similar manner to the process of developing a Lagrangian objective function in standard optimization techniques.

The necessary conditions for the optimization of the path planning problem are developed from the integral of the standard Euler-Lagrange equation for the calculus of variations problem. These equations are derived directly from a Taylor series expansion of

the unconstrained minimization and are then approximated by finite difference equations to allow for a numerical solution. A Newton-Raphson method is used to determine the roots to the set of finite difference equations. The expected form of the theoretical solutions is then described to form a basis of comparison to computer generated solutions.

Test cases are shown demonstrating the feasibility of the algorithm in two- and three-dimensional space. In general the algorithm that has been developed works well for the class of problems involving small numbers of circular or spherical obstacles. However, as the complexity of the obstacle geometry increases the algorithm becomes unstable. This is caused by an increase in the round-off error generating parasitic solutions. These parasitic solutions cause paths to be generated which violate the obstacle boundaries and thus generate invalid paths. It is recommended in future research that alternative methods be studied for solving the system of first order differential equations to eliminate this problem. Investigating possible modifications to the objective function by using appropriate weighting factors on the energy and slack functions may also enhance the potential for eliminating the instabilities.

CHAPTER 1
AN INTRODUCTION TO PATH PLANNING AND DIFFERENTIABLE MANIFOLDS

1.1 Introduction

A current trend in the the automation of both traditional manufacturing processes and navigation of unmanned vehicles is towards developing methodologies for self-control of machines in a working environment, independent of human supervision. The tasks involved in these methodologies include gathering information about the environment using vision and other types of sensors, assimilating this data in some useful manner, and finally acting upon this information to achieve an objective which is in accordance with its overall function. Navigation, in a broad sense, is one of the activities which falls under the last of these tasks. It involves taking known information about an environment and requires acting upon this data to plan and move along a path through a machine's environment consistent with its objective. This objective usually involves getting from point A to point B while avoiding all known obstacles and is equally applicable to a wide range of mechanical operations, from moving a robot end effecter to guiding an unmanned vehicle.

In traditional navigation methodology the task is normally divided into two steps. Firstly, a path planning algorithm generates an idealistic path based on what is known about the environment, without taking into account the limitations of the machine. Secondly, a path following algorithm is implemented which is specific to the type of machine under consideration, and which allows the machine to follow the desired path as closely as possible. It is the first of these navigation steps, the path planning problem, which is investigated in this dissertation. This problem can be simply stated as one of finding a path from point A to point B which avoids all known obstacles. Usually some criteria are established for choosing a path from among all feasible paths in the problem.

Path planning usually involves moving points, rigid bodies, or collections of rigid bodies in the case of robot manipulators, along collision free paths among obstacles. Points and rigid bodies moving in Euclidean space have an underlying mathematical structure of a differentiable manifold, which allows for an elegant analytical description of the problem. Most of the methods developed for path planning have been concerned with finding ways to approximate this manifold in a data structure and to allow for a computational scheme to use the data structure to find a path. The method proposed and developed in this dissertation is based on the analytical tools of variational calculus and its role in finding paths of minimum length in a manifold. Due to recent advances in computational methods, it is now possible to extend variational calculus into the realm of computers and therefore apply these tools in developing algorithms for path planning.

In this chapter the general concept of a configuration space and its relationship to differentiable manifolds will be developed. Section 1.2 will establish the mathematical context of the problem. In Section 1.3, the existing path planning methods are organized based on their methods for approximating the mathematical concept of a differentiable manifold. This will lead to a justification for researching a new method in Section 1.4.

1.2 Differentiable Manifolds

All path planning algorithms are developed in the context of a configuration space. A configuration space is an analytical representation of a physical problem as a point in the space of its variables. As an example, a planar rigid body can be represented by three independent variables X,Y, and α , its physical space. The body can then be mapped to a point in its configuration space as shown in Figure 1.1, with a suitable restriction on α of an interval of $[0,2\pi)$.

It is important to note that the dimension of the configuration space is equal to the number of independent parameters needed to describe the problem. For example, the planar rigid body could also be described by two points $A(X_a,Y_a)$ and $B(X_b,Y_b)$, of which three coordinates are independent, and then be represented in a slightly different

configuration space(X_a,Y_a,X_b) or (X_a,Y_a,Y_b). The rigid body can be represented in many ways, for example, by homogeneous coordinates or quaternions. However, none of these coordinate systems or charts, as they are called in differential geometry, alters the underlying behavior of the planar rigid body and the essential information about the body's position and orientation. It is intuitively obvious that all the coordinate systems must describe this position and orientation in some equivalent manner and that there is an equivalence relationship between any two coordinate systems. In this example the differential manifold describes the underlying concept of the rigid body and the equivalence of all of its associated charts.

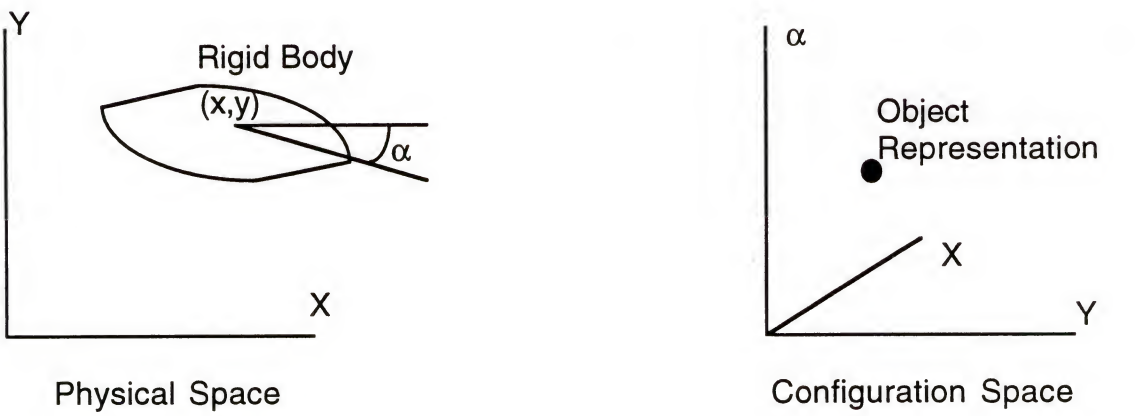


Figure 1.1 Two Representations of a Rigid Body

In mathematical terms a general manifold is an ordered pair (M,*S*) of the point set M, describing the configurations of the physical system as a set of points, and a set of equivalence relationships *S* relating the different charts that can represent those points. The equivalence relationships rely on the conditions that each chart maps a region of M into R^n and that there are smooth functions called diffeomorphisms which will map coordinates

from chart to chart. These functions are developed independently of coordinate systems and allow differentiable manifolds to describe the underlying behavior of a physical system as well as the relationships between the diffeomorphisms. A diagram showing the basic relationships between manifolds and their mappings is given in Figure 1.2. Open subsets U and V of the point set M can be mapped into open subsets of R^n given by A and B.

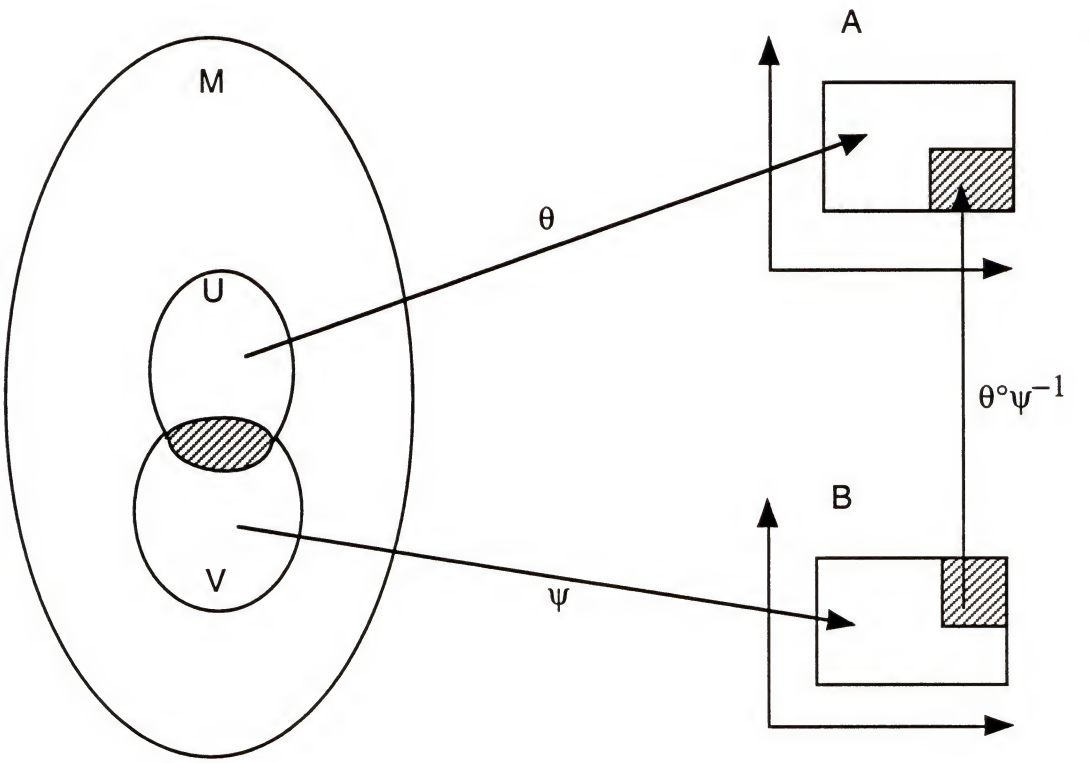


Figure 1.2 Representation of a Differential Manifold and Its Mappings

The ordered pairs (U,θ) and (V,ψ) are charts and their inverse mappings (A,θ^{-1}) and (B,ψ^{-1}) are parameterizations. Finally, the mapping from A to B is given by $\theta^{\circ}\psi^{-1}$, which is called a diffeomorphism showing the equivalence between the intersection of the regions in the point set M.

The configuration space is a differentiable manifold with a choice of coordinate systems, or more specifically a chart in the atlas of the differentiable manifold. For the purpose of this research a reasonable coordinate system will be chosen to represent the physical problem and the terms configuration space and manifold will be used interchangeably. It will be understood that the results can be generalized to other coordinate systems with a suitable mapping. The path planning problems considered in this thesis will involve moving a point among obstacles in a Euclidean space.

The next task is to show how points and rigid bodies can be mapped into points in a multi-dimensional differentiable manifold. The key elements of these proofs will be argued intuitively, based on a knowledge of the geometry of points and rigid bodies. More complete proofs are given in Robot Motion Planning [Lato 91] by Latombe and Manifolds and Mechanics [Jone 87] by Jones, Gray, and Hutton. In the case of a point described by a Euclidean space R^n, it is trivial to state that any region of R^n can be mapped into any other region of R^n and that Euclidean space is a point set. Any coordinate system for R^n can be rotated and translated in a smooth manner to generate another coordinate system, and this allows for a set of equivalence relationships to be described.

The proof that a rigid body can be represented by a differentiable manifold is more complicated. The manifold describing a rigid body is R^nx$SO(n)$, where R^n is the space of translations and $SO(n)$ is the special orthogonal group describing the space of rotations of the rigid body. That this space is a continuous point set is straightforward, and can been seen as a generalization of the configuration space in Figure 1.1. It now remains to show that each neighborhood of a point in the manifold R^nx$SO(n)$ can be smoothly mapped into a region of R^n. This is closely related to the first order kinematic properties of rigid body motion. In the neighborhood of a point, describing a configuration of the rigid body, all other points can be closely approximated by infinitesimal displacements. These infinitesimal displacements are linear or affine mappings and the approximation of any point in the neighborhood is an affine approximation. The space that represents linear or

affine approximations is R^n. Thus a neighborhood of a point in the manifold will map to R^n. The smoothness of the mapping from R^nxSO(n) to R^n comes from the property of being able to establish derivatives of the motion at any given point.

It has now been established that points and rigid bodies can be represented as points in a differentiable manifold. However, in order to use manifolds as a model for path planning it is necessary to map obstacles from the physical space into the configuration space. The importance of mapping obstacles into the configuration space is that any path planning problem can then be reformulated as moving a point through an n-dimensional configuration space while avoiding obstacles. Lozano-Perez[Loza 83] has developed a method for mapping convex polygonal obstacles and bodies from the physical space to the configuration space. Lozano-Perez discusses how this method can be used to model robot arms coming into contact with obstacles and this has been successfully implemented by Lumelsky[Lum1 85],[Lum2 85],[Lum3 85]. Lozano-Perez' method will be described here to give a conceptual overview of how polygonal obstacles are mapped from the physical to configuration spaces and how to extend this method for any type of obstacle.

The configuration space for a planar rigid body has previously been described in Figure 1.1. Each slice of the space parallel to the X-Y plane has a value of constant α. Therefore, when a body comes in contact with an obstacle at a fixed angle it will trace some shape around that obstacle. For example when a triangular body with a fixed orientation α, comes in contact with a convex obstacle it is possible to trace out a polygonal shape as shown in Figure 1.3. It is important to note that the boundaries of the new irregular polygon are made up of displaced edges of the obstacle B or the body A. A series of irregular polygons can then be generated by varying α from $[0,2\pi)$. The irregular polygons can be stacked in order of increasing α to generate the mapping of the obstacle in the configuration space where it is bounded by ruled surfaces as shown in Figure 1.4.

Lozano-Perez has shown that this algorithm can be extended for nonconvex obstacles by simply decomposing these obstacle into a series of overlapping, convex obstacles. The algorithm can be further extended to a space of arbitrary dimension.

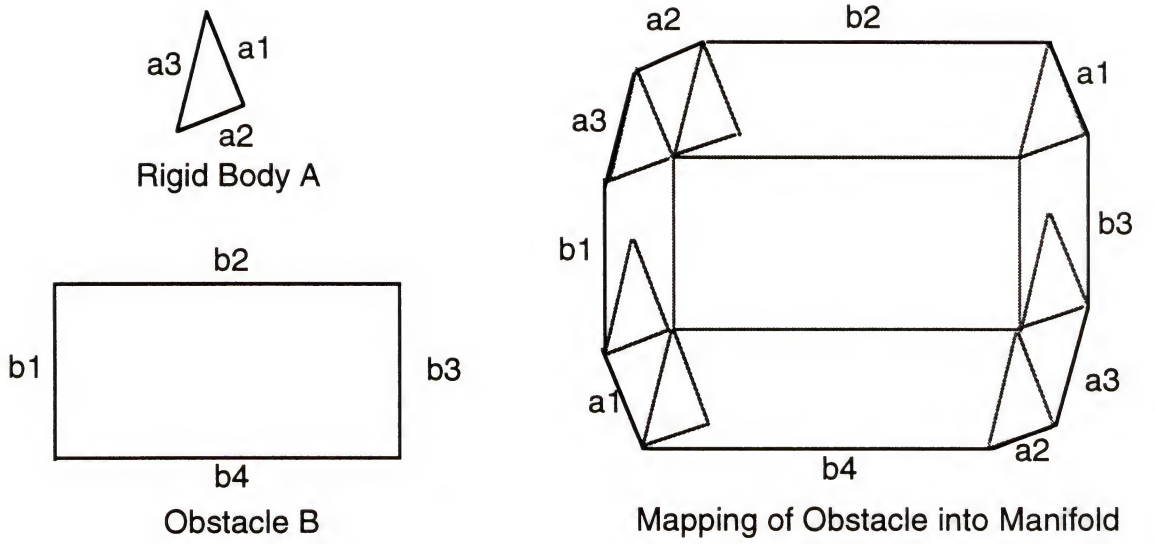


Figure 1.3 Mapping of an Obstacle into a Slice of a Manifold

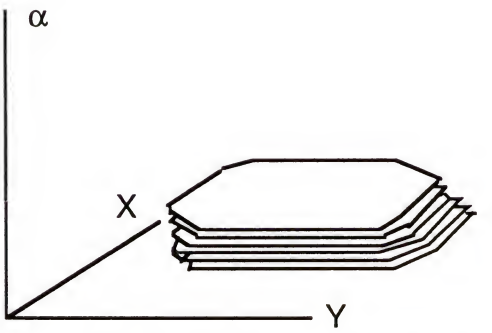


Figure 1.4 An Obstacle Mapped into the Configuration Space

These methods work for polygons in the plane and convex hulls in three-dimensional space. However, if the body or obstacle is represented by a series of algebraic curves or surfaces, algorithms exist for approximating these curves or surfaces by convex polyhedra. In this way, more general representations of bodies or obstacles can be modified to allow for use in the Lozano-Perez algorithm.

1.3 Review of Current Path Planning Methodologies

It was shown in the previous section that the path planning problem, consisting of a point or body being moved and the obstacles to be avoided, can be converted into a general problem of moving a point among obstacles in a manifold. It remains to describe the current methods for converting this problem into a computational scheme. The first group of path planning methods described in this section are labeled as decomposition methods, because they sample the configuration space in a variety of ways and decompose it in a suitable manner so that it can be stored in a data structure. Path planning via network decomposition methods can be seen as a descretization of the configuration space into a network combined with an efficient search to find the path. There are many variations on this idea, but only two basic methods will be described here as being representative. What all the methods have in common is that they preserve the connectivity of the configuration space around the obstacles. For instance if two nodes of a data structure, representing two regions of the configuration space, are connected to each other, then it must be possible for a point to move from one region to another without violating the boundary of an obstacle.

The first method represent points in the configuration space as nodes in a network. The nodes represent the start and goal position and the vertices of the polygonal obstacles. The connections in the network represent points visible from a given point as shown in the visibility graph in Figure 1.5. The time to generate the graph is $O(n^2 \log n)$, where n is the number of vertices in the problem. Each connection is weighted by its length and a search is done using the A* search algorithm [Hart 68], which finds an optimal path in $O(n^2)$.

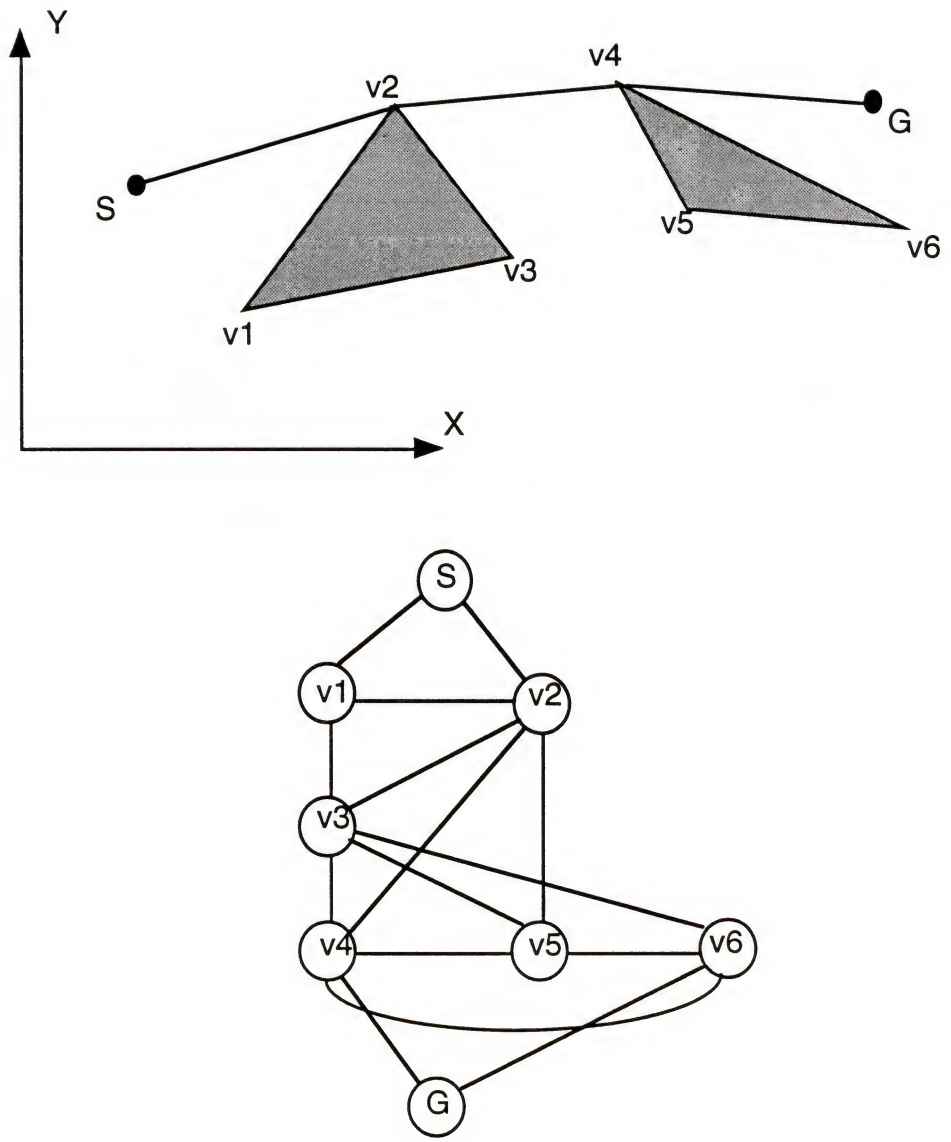
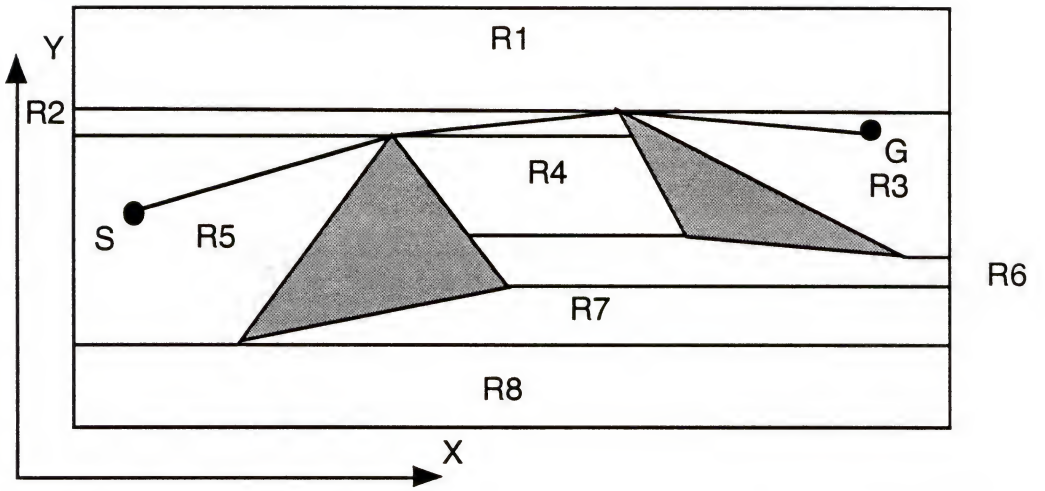


Figure 1.5 Decomposition of Planar Polygons to a Visibility Graph

A second strategy has been to decompose the work space into a collection of cells which may or may not overlap. Each cell represents a region of the configuration space which does not contain an obstacle.The connectivity of the nodes in this graph represents the relationship between adjacent cells. One such method has been developed by Ahrikencheikh [Ahri 94] in which each cell is a slice of the configuration space. For the case of a point moving in the plane, it takes $O(n \log n)$ to generate the network where n is the total number of edges of the obstacles. An example of this type of representation is shown in Figure 1.6. The optimal search, which accounts for the velocity, acceleration and clearance constraints, can be done in $O(n^4)$ time.

The extension from the plane into higher dimensions for the network decomposition methods poses significant problems in computing efficient paths. For instance, the shortest length path between opposing vertices of a cube which are not visible to each other, crosses two faces of the cube and does not contain any other vertices. Therefore searching a visibility graph would not lead to the shortest path. This problem can be generalized to an example where the shortest length path does not contain the vertices of any of the obstacles. A similar problem exists in determining cells or volumes in the second network decomposition problem. These problems have been studied and can be overcome using such methods as unfolding polyhedral obstacles into a plane (see [Ahri 94]). In general, decomposition methods are only used for two or three dimensional problems, as the added computations required to simplify higher dimensional problems render the algorithms inefficient.

A second approach to path planning has been to apply optimal control to finding a path among known obstacles. The configuration space description for the problem adds the time derivatives of the spatial coordinates as dimensions of the configuration space. These methods combine the the tasks of path planning and trajectory following by considering the dynamic equations of the robot arm as well as the motor torques required to generate the motion. An optimal control problem is formulated and solved using dynamic programming by Shin and McKay [Shin 85].



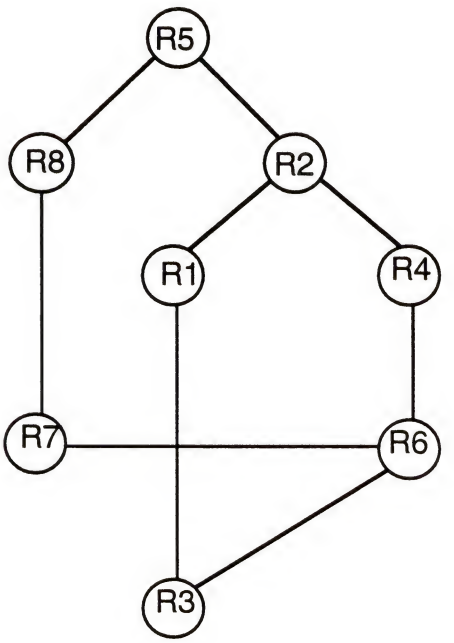


Figure 1.6 Grid Decomposition of Planar Polygons

The potential field method of path planning models the goal position as an attracting force field and each obstacle as a repelling force field as shown in Figure 1.7. Each point in the configuration space is then assigned a value as the sum of all the individual fields. The rigid body, represented by a point, slides downhill to the next available position with a lower potential. The disadvantage of this technique is that the point traveling through the field can become stuck at some intermediate position where it is locally the point of lowest potential. The paths taken by the point in the configuration space is dependent on the arbitrarily chosen potential functions. This method was developed by Kahtib [Khat 86].

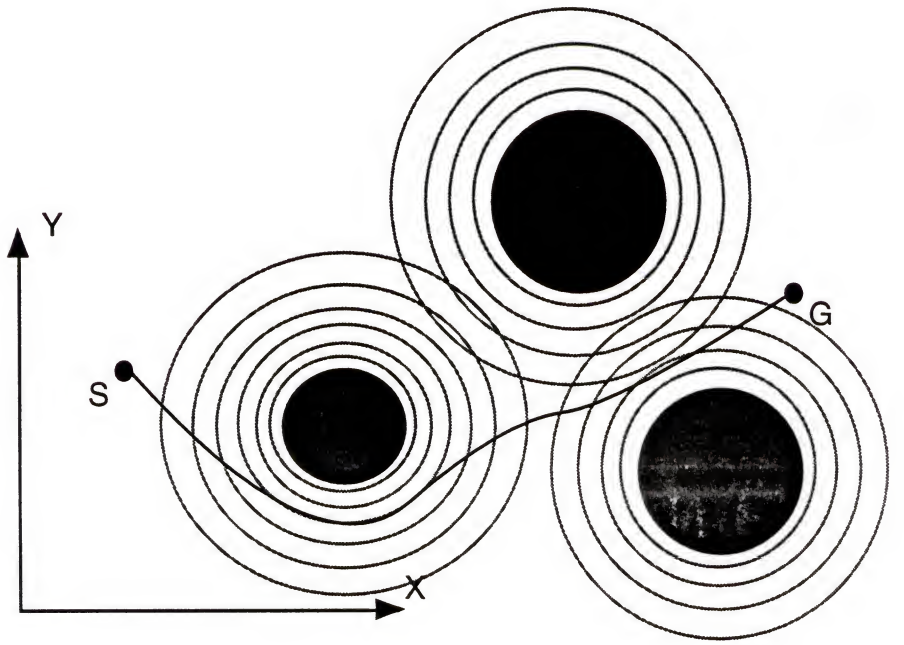


Figure 1.7. Path Planning Using Potential Fields

The last method described in this section is associated with the mathematical analysis of singularities. A school of mathematical research in Russia, lead by V.I. Arnol'd, has conducted research into the problem of bypassing an obstacle. Although the

research generated on this topic is not generally associated with current path planning methodologies, it lends important insight into the mathematical structure of the manifolds used to describe path planning. The basis of their methods is to use a Riemannian metric and calculus of variations to solve the path planning around a single obstacle. This is the same mathematical basis as the work developed in this thesis. However, the aim of their research leads in a different direction and is not focused towards developing a computer algorithm or for path planning with more than one obstacle.

Singularity theory is the study of discontinuities in the mapping of functions from R^n to R^m. For instance in Figure 1.8 if the starting point for a minimum length path is fixed and the end point is allowed to vary, the path can be considered as a function of two variables.

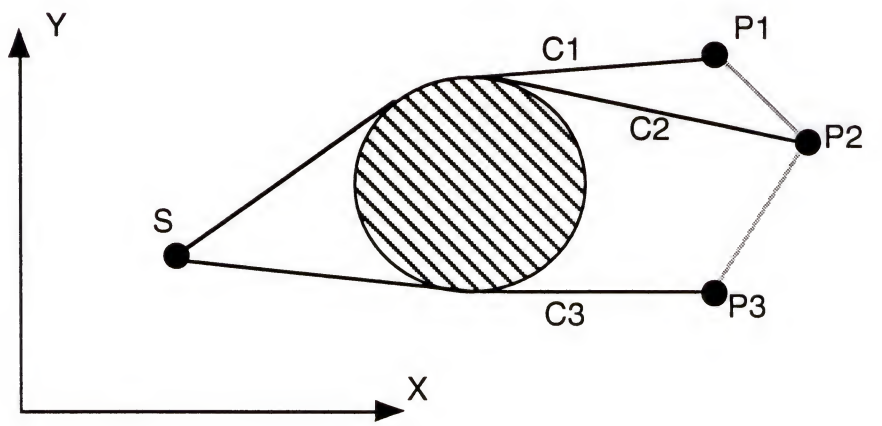


Figure 1.8. Path Discontinuities for a Smooth Change in Endpoint.

If the end point moves from P_1 to P_2 around the obstacle in a continuous manner the curve will smoothly deform from C_1 to C_2. However, if the end point moves from P_2 to P_3 there will be a jump discontinuity from C_2 to C_3. It is the study of these discontinuities and their classification which is the aim of Singularity theory. Representative papers on this topic

include "Singularities in Variational Calculus"[Arno 83], by V.I. Arnol'd, "Singular Lagrangian Manifolds and their Lagrangian Maps"[Give 90] by A.B. Givental, and "Wavefronts and reflection groups"[Shch 88] by O.P. Shcherbak.

1.4 Justification for a New Method

A brief review of existing methods of path planning has been given in the previous section by describing the different approaches undertaken to approximate the path planning problem. In Section 1.2 the mathematics of path planning was shown to be that of differentiable manifolds, which provides many topological and analytical tools. For the most part these tools are not sufficiently utilized in existing path planning methodologies. The algorithms reviewed so far have either made arbitrary assumptions about how to approximate the configuration space, have introduced a state space formulation to allow for computation of motor torques, or require the user to create arbitrary field functions. All of these different approaches tend to obscure the underlying structure of the problem, which provides sufficient information for generating minimum length paths. The field of calculus of variations provides the necessary analytical tools for finding shortest length paths on manifolds. However, to the best of the author's knowledge, variational calculus has not been used as the basis for a general path planning algorithm. The work in singularity theory uses the same mathematical basis, but their goal is not to develop a computer algorithm for use with many obstacles.

The question then to be asked is, if analytical methods for finding minimum length paths on manifolds exist, why have these tools not been applied to path planing? A possible answer to this question is that while the calculus of variations succinctly represents the path planning problem, it has been virtually impossible to solve these problems except for the simplest cases and no general algorithms have existed for use on computers. Over the past few years research into general algorithms for these problems have been developed culminating in the book Constrained Optimization in the Calculus of Variations and Optimal Control Theory [Greg 92], by John Gregory and Cantian Lin.

Therefore, at the present time, it seems worthwhile to investigate the application of these methods to path planning to determine their feasibility and to allow for a comparison between existing techniques and path planning through the use of calculus of variations.

In Chapter 2 the analytical basis for variational calculus and its application to path planning is developed. It will be shown how the calculus of variations can be used to succinctly represent the general path planning problem as a constrained optimization and that this problem takes the form of the Problem of Bolza. It will then be shown how this constrained problem can be converted into an unconstrained optimization and that critical solutions for the problem can be found using the Euler-Lagrange equations.

In Chapter 3, an algorithm is developed for solving the integral of the Euler-Lagrange equations. This discussion is specifically tailored for the path planning problem. The expected form of the solution is described to give an indication of how the algorithm should behave. Two tests are studied and a modification to a stage wise optimization by adding one obstacle at a time is proposed to improve the robustness of the algorithm.

Chapter 4 gives a detailed analysis of the algorithm for a point moving in the Euclidean plane. The strengths and weaknesses of the basic algorithm are explored through a variety of test cases. The discussion of the expected form of the solution is expanded into a qualitative test to judge the merit of the path that is found. When the algorithm fails to generate a valid path, a modified form of the algorithm is tested using stage wise optimization. In the final section test cases are developed for studying path planning around moving obstacles.

In Chapter 5, the algorithm is extended into three-dimensional space. A series of examples are shown which are more illustrative in nature demonstrating the extendibility of the algorithm into three-dimensional space. In general the basic algorithm has a more difficult time converging based on the form of the solutions that are generated.

Chapter 6 provides a summary of the tests and suggests reasons for the failure in certain instances and suggests possible modifications. The difficulties in obtaining sufficiency conditions and extension of the algorithm to other norms completes the discussion of the research that has been conducted.

CHAPTER 2
FORMULATION OF THE PATH PLANNING PROBLEM USING THE CALCULUS OF VARIATIONS

2.1 Formulation of the Path Planning Problem

In the previous chapter an overview of path planning methods was introduced and, in a general form, it was shown how these problems can be modeled by differentiable manifolds. It was further explained how these manifolds have a strong analytical foundation and it was proposed that variational calculus could be used as a basis for a new formulation of the path planning problem. The methodology developed in this chapter differs from existing techniques in that the problem is described analytically and utilizes an optimization as opposed to an approximation of the problem combined with a search through a data structure. There are strong similarities between standard optimization techniques and optimization using the calculus of variations which will be exploited throughout the mathematical development of the path planning problem in the rest of this chapter.

As a preliminary step it is necessary to show how the length of a parametrized curve segment is measured. In the simplest geometric case given a vector V from a point p_a to a point p_b, the distance in a Euclidean space between these two points can be calculated by the equation,

$$d(p_a,p_b) = || V|| =[v_1v_1 + v_2v_2 + \cdots + v_nv_n]^{1/2} \qquad (2.1),$$

using the Euclidean norm. In general, for two vectors X and Y, and a scalar a, a norm is a function which has the following properties,

1. $\|X\| > 0$, for all $X \neq 0$
2. $\|X\| = 0$, if $X = 0$
3. $\|a*X\| = |a|*\|X\|$, $a \varepsilon R$
4. $\|X+Y\| \leq \|X\| + \|Y\|$.

For path planning the important properties of the function are that it is positive definite and that it passes through the origin. The path planning problem is concerned with the minimization of curves and it is therefore necessary to generalize this concept of distance to quantify the length of a curve segment.

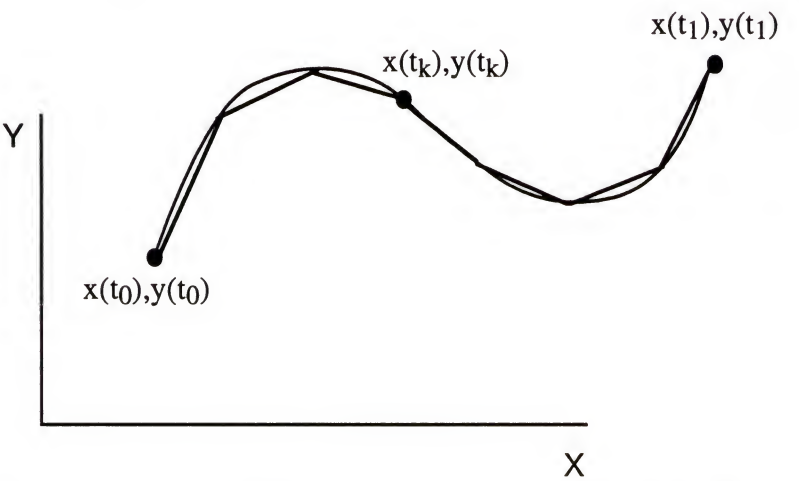


Figure 2.1 Approximation of a Curve by Straight Line Segments

The curve segment given in Figure 2.1 can be approximated by straight line segments. The length of each segment can be calculated using equation (2.1) above. It is straightforward to sum the length of these segments to obtain an approximate length of the curve and then to refine the approximation by using more line segments. This leads to a limiting process in analytical calculus from which the the length can explicitly be given by the integral equation,

$$L_a^b = \int_a^b \left\| \frac{dx}{dt} \right\| dt = \int_a^b [\dot{x}_1^2 + \dot{x}_2^2 + \cdots + \dot{x}_n^2]^{1/2} \, dt \tag{2.2}$$

The limiting process is invariant with a change of parametrization of the curve. The refinement of the point approximation transforms the vectors into tangent vectors as the limit is reached and the length of the curve segment becomes the integral of the norm of these tangents.

In differential geometry the norm is defined as the square root of the metric which for the type of problem under consideration is the Euclidean metric given by

$$\langle v,v \rangle = [v_1 v_1 + v_2 v_2 + \cdots + v_n v_n] \tag{2.3}$$

The metric is similar to the norm in that it is positive definite, zero valued when the vector is a zero vector and invariant with a change in parametrization. In a similar manner for the method used to obtain the length of a curve segment, an energy function can be defined which is based on the metric and is given by

$$E_a^b = \frac{1}{2} \int_a^b \left\langle \frac{dx}{dt}, \frac{dx}{dt} \right\rangle dt = \frac{1}{2} \int_a^b [\dot{x}_1^2 + \dot{x}_2^2 + \cdots + \dot{x}_n^2] \, dt \tag{2.4}$$

The energy function is the integral of the Euclidean metric of the tangent vectors. These tangent vectors define a one dimensional tangent space at each point on the curve. For general differentiable manifolds the Euclidean metric is generalized to a Riemannian metric, which is any positive definite inner product over the tangent spaces of the manifold. It is important to note that the metric is arbitrarily defined for the tangent spaces on a manifold and the tangent space and the dimension of the manifold are the same size. Once this metric is chosen the norm is automatically derived with the properties listed above and the concept of distance in the space is then subsequently defined.

To apply the concepts of length and energy of a curve segment to the path planning problem, an example will be formulated for a point moving in two-dimensional space between circular obstacles. The mapping of a point from the physical space to the configuration space is trivial and the representation in either space is identical. The path planning problem can then be extended into N dimensions with M number of obstacles in a straightforward manner.

In Figure 2.2, a simple example is shown involving a start position, a goal position, and two obstacles. It is desired to move from the start to the goal position along a path of minimum length, while avoiding these obstacles. The approach is to develop this optimization problem in a manner suitable for solution using calculus of variations methods. The path shown in Figure 2.2 is a piece wise smooth curve of minimum length from the start to the goal position. The purpose is to define the objective function which represents the length of any curve in the space which satisfies the boundary conditions and to then minimize this function to determine the optimal path.

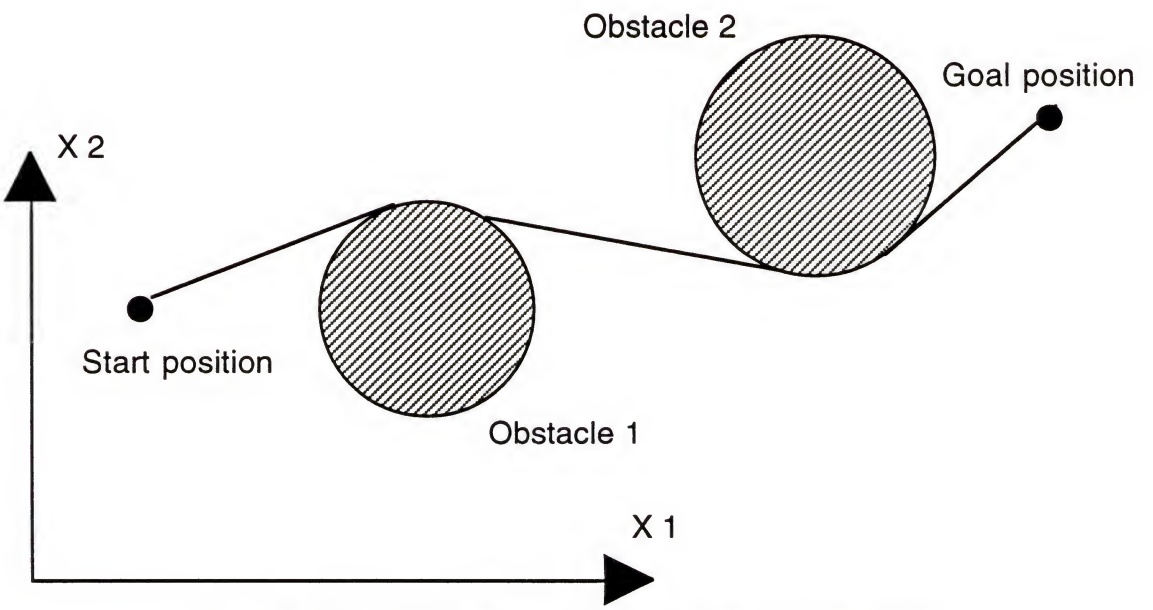


Figure 2.2 A Two-Dimensional Path Planning Problem

The length of the path can then be found by using equation (2.2) for the planar case,

$$L = \int_0^1 [\dot{x}_1^2 + \dot{x}_2^2]^{1/2}\, dt \qquad (2.5).$$

The parametrization of the curve has been normalized so that the endpoints of integration range from [0,1]. To simplify calculations by eliminating the square root in equation (2.5), the energy function can be minimized over the same curve segment. A discussion of the equivalence of minimizing the energy function E and the length L of the path will be postponed until the end of this chapter. At the present it is sufficient to point out that the equivalence is based on the shared properties of the norm and the energy functions and that a stationary curve for one of these functions is automatically a stationary curve for the other. The energy function for the planar curve is given by

$$E = \int_0^1 [\dot{x}_1^2 + \dot{x}_2^2]dt \qquad (2.6).$$

The 1/2 in front of the energy function is omitted because it has no bearing on the optimization. To return to the path planning problem it is necessary to minimize the energy function while avoiding the obstacles. This can be stated as follows:

$$\text{minimize: } E = \int_0^1 [\dot{x}_1^2 + \dot{x}_2^2]dt \qquad (2.7a)$$

where

$$x_1(0) = x_{1\ \text{start position}} \qquad x_1(1) = x_{1\ \text{goal position}} \qquad (2.7b)$$

$$x_2(0) = x_{2\ \text{start position}} \qquad x_2(1) = x_{2\ \text{goal position}} \qquad (2.7c)$$

subject to avoiding the obstacles,

$$\psi_1 = -\{(x_1 - a_1)^2 + (x_2 - b_1)^2 - r_1^2\} \le 0 \qquad (2.7d)$$

$$\psi_2 = -\{(x_1 - a_2)^2 + (x_2 - b_2)^2 - r_2^2\} \le 0 \qquad (2.7e).$$

The path planning problem can now be extended for the more general case of N dimensions with M obstacles;

$$\text{minimize: } E = \int_0^1 [\dot{x}_1^2 + \dot{x}_2^2 + \cdots + \dot{x}_N^2]dt \tag{2.8a}$$

where,

$$\begin{aligned} x_1(0) &= x_{1\text{ start position}} & x_1(1) &= x_{1\text{ goal position}} \\ x_2(0) &= x_{2\text{ start position}} & x_2(1) &= x_{2\text{ goal position}} \\ &\vdots \\ x_N(0) &= x_{N\text{ start position}} & x_N(1) &= x_{N\text{ goal position}} \end{aligned} \tag{2.8b},$$

subject to avoiding the obstacles

$$\begin{aligned} \psi_1 &= -\{(x_1 - a_{11})^2 + (x_2 - a_{12})^2 + \cdots + (x_N - a_{1N})^2 - r_1^2\} \le 0 \\ \psi_2 &= -\{(x_1 - a_{21})^2 + (x_2 - a_{22})^2 + \cdots + (x_N - a_{2N})^2 - r_2^2\} \le 0 \\ &\vdots \\ \psi_M &= -\{(x_1 - a_{M1})^2 + (x_2 - a_{M2})^2 + \cdots + (x_N - a_{MN})^2 - r_M^2\} \le 0 \end{aligned} \tag{2.8c}.$$

Equation (2.8) is now a formulation of a boundary value, calculus of variations problem employing inequality constraints and is a modified version of the Problem of Bolza. The Problem of Bolza was first developed for minimizing the area of surfaces of revolution subject to equality constraints. The critical point solutions were first developed by G. A. Bliss and were later extended to include inequality constraints by F. A. Valentine (see [Greg 92]).

2.2 Conversion of the Problem of Bolza to an Unconstrained Problem

It is necessary to simplify the path planning problem represented by equations (2.7) and (2.8) down to one unconstrained equation which can then be optimized. The techniques used to accomplish this are similar to those used in optimization for a

discrete number of variables, namely, convert inequality constraints to equality constraints by the introduction of slack variables, and then combine them with the objective function using Lagrange multipliers. The method proposed by Gregory and Lin [Greg 92] follows along these lines with the modification that the slack variables and the Lagrange multipliers become functions of the independent parameter t. The equations (2.7) can now be reformulated with no constraints where the function under the integral becomes,

$$F(t,X,\dot{X}) = \dot{x}_1^2+\dot{x}_2^2 - \dot{x}_3\{(x_1- a_1)^2 + (x_2- b_1)^2 - r_1^2 - \dot{x}_4^2\} - \dot{x}_5\{(x_1- a_2)^2 + (x_2- b_2)^2 - r_2^2 - \dot{x}_6^2\} \quad (2.9).$$

The functions x_3 and x_5 are integrals of the Lagrange multiplier functions and x_4 and x_6 are integrals of the slack variable functions. The reformulated problem is to find a solution vector

$$X(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \end{pmatrix} \quad (2.10a)$$

which is a stationary point of the integral

$$I(x) = \int_a^b F[t,X(t),\dot{X}(t)]dt \quad (2.10b),$$

where the boundary conditions are given by

$$X(0) = \begin{pmatrix} x_1\text{start} \\ x_2\text{start} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (2.10c)$$

and

$$x_1(1) = x_1goal$$
$$x_2(1) = x_2goal \qquad (2.10d).$$

It should be noted that the ending boundary condition given in equations (2.10d) is not completely specified. This leads to the necessity of using transversality conditions which will be discussed later in the chapter. The restriction on the individual functions of (2.10a) are that they are piece wise smooth which in turn allows the derivatives to be piece wise continuous. A function is piece wise continuous if it is continuous over an interval except at a finite number of points where there are jump discontinuities. Finite limits exist on both sides of these discontinuities. By setting the Lagrange functions and slack functions to be derivatives, they are then piece wise continuous, which is the type of behavior needed to allow for the complimentary slackness conditions, namely

$$\dot{x}_3\dot{x}_4 = 0 \qquad (2.11)$$

and

$$\dot{x}_5\dot{x}_6 = 0 \qquad (2.12).$$

Each of the functions $x_3(t)$, $x_4(t)$, $x_5(t)$, and $x_6(t)$ is only specified in terms of its derivative. To uniquely define each function the constant of integration is set to zero in the boundary condition (2.10c).

The general path planning problem can be transformed into an unconstrained problem in a similar manner for N dimensions and M obstacles. The function inside the integral becomes,

$$F(t,X,\dot{X}) = \sum_{i=1}^{N} \dot{x}_i^2 - \sum_{j=1}^{M} \dot{x}_{2j-1+N}\{\sum_{k=1}^{N} (x_k - a_{jk})^2 - r_j^2 - \dot{x}_{2j+N}^2\} \qquad (2.13),$$

where a_{jk} is the k^{th} coordinate of the center of the j^{th} obstacle, and the solution vector then becomes

$$X(t) = \begin{pmatrix} x_1(t) \\ \vdots \\ x_N(t) \\ \\ x_{N+1}(t) \\ x_{N+2}(t) \\ \vdots \\ x_{N+2M-1}(t) \\ x_{N+2M}(t) \end{pmatrix} \qquad (2.14a).$$

The integral to be minimized is

$$I(x) = \int_0^1 F[t,X(t),\dot{X}(t)]dt \qquad (2.14b),$$

and the boundary conditions are

$$X(0) = \begin{pmatrix} x_1 start \\ \vdots \\ x_N start \\ 0 \\ \vdots \\ 0 \end{pmatrix} \qquad (2.14c),$$

and

$$\begin{aligned} x_1(1) &= x_1 goal \\ &\vdots \\ x_N(1) &= x_N goal \end{aligned} \qquad (2.14d).$$

2.3 Taylor Series Expansion of the Integral Equations

In the previous section the path planning problem was converted from a constrained optimization to an unconstrained problem. The next step is to find a method for solving this type of minimization. In general, variational calculus is concerned with finding a solution curve or set of curves which optimizes an integral function. In the path planning problem, it is desired to minimize the integral in equation (2.14b) subject to the boundary conditions, (2.14c) and (2.14d). The next step is to determine the analytical methods for finding functions that provide stationary curves to the integral equation. This is analogous to techniques used in finding stationary points for a standard optimization.

The following derivation is the classical method for determining the Euler-Lagrange equation as a condition for a stationary point in the calculus of variations. Some of the details of the derivation, regarding theorems from analytical calculus, have been omitted so that the similarities between optimization using calculus of variations and standard optimization techniques for a discrete set of variables are made clearer.

In general, the calculus of variations problem is concerned with finding a set of optimal functions

$$x_o = x_o(t) \tag{2.15a}$$

such that,

$$I(x) = \int_a^b f[t,x(t),\dot{x}(t)]dt \tag{2.15b}$$

is minimized where,

$$x(a) = a \text{ and } x(b) = b, \tag{2.15c}$$

where in the path planning problem x is a vector,

$$x = (x^1(t),x^2(t),...x^m(t))^T. \tag{2.15d}.$$

The method for solving this problem is to assume that a solution curve $x_o = x_o(t)$ exists and to examine candidate solutions about the curve x_o. Any candidate curve $x = x(t)$, in the neighborhood of the solution can be expressed as the sum of two curves, $x = x_o(t) + \varepsilon y(t)$, where ε is a scalar, as shown in Figure 2.3.

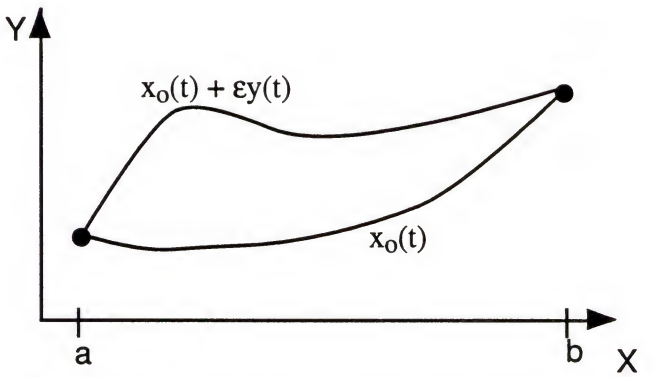


Figure 2.3 Variations Around a Solution Curve

Restrictions are placed on y(t), such that it is zero valued at its end points to meet the boundary conditions and its first derivative exists. It is important to point out that y(t) is an arbitrarily chosen infinitesimal change. The difference between x(t) and $x_o(t)$ is called the variation and is expressed by $\varepsilon y(t)$. Equation (2.15b) can then be reformulated as a function of a single variable, whose stationary point is found when ε is set equal to zero,

$$F(\varepsilon) = \int_a^b f[t, x_o(t) + \varepsilon y(t), \dot{x}_o(t) + \varepsilon \dot{y}_o(t)]dt \qquad (2.16).$$

The next several steps follow closely the method used for finding stationary points in standard optimization techniques. Equation (2.16) can be expanded in a Taylor Series about the point ε equal to zero, giving a first order approximation to the integral,

$$F(\varepsilon)_{\varepsilon=0} = F(0) + \varepsilon \int_a^b [f_x y + f_{\dot{x}} \dot{y}]dt + O(\varepsilon^2) \qquad (2.17).$$

To find a stationary point it now remains to set the first derivative to zero and solve,

$$F'(0) = I'(x,y) = \int_a^b [f_x y + f_{\dot{x}} \dot{y}]dt = 0 \qquad (2.18)$$

By integrating equation (2.18) by parts and simplifying the terms, the standard Euler-Lagrange equation is derived

$$\frac{d}{dt}(f_{\dot{x}}) - f_x = 0 \qquad (2.19),$$

which gives the conditions for a stationary point for the calculus of variations problem described by equations (2.18) through (2.21). In optimization terminology only the necessary condition has been considered and from a theoretical point of view only a stationary point has been determined. No consideration has been given as to whether the stationary point is a maximum, minimum, or an inflection point.

The Euler-Lagrange equation gives the conditions for the stationary point of the integral in the form of a second order, nonlinear differential equation when a single curve x_0 is being determined and a set of coupled, second order nonlinear differential equations when x_0 is a set of curves being determined. These have to be integrated twice to determine the stationary curve or curves. It is straightforward to see that the solution of these differential equations becomes more difficult as the complexity of the problem increases. It should be evident that many problems that can be formulated using variational calculus cannot easily be solved. Gregory and Lin [Greg 92] have developed a method around this problem which is suitable for computer solutions as presented in Chapter 3.

2.4 Transversality Conditions

The unconstrained formulation of the path planning problem has both fully described boundary conditions for the physical functions, and partially described boundary conditions for the slack and Lagrange functions. It now remains to modify the Euler-Lagrange equation to account for the incomplete boundary conditions. In Figure 2.2 it was shown that the curve y(t) had to be zero valued at its boundaries. This condition is now relaxed at b because the value of the functions are unknown at this boundary as shown in Figure 2.4. The derivation follows along the same lines as that in the previous section until it comes to integrating equation (2.16) by parts. The right hand boundary of the variation y(t) is unknown and integrals containing these terms cannot be canceled. These terms are collected into the transversality condition which replaces the boundary condition y(b) yielding

$$f_{\dot{x}}(b,x_o(b),\dot{x}_o(b)) = 0 \qquad (2.20).$$

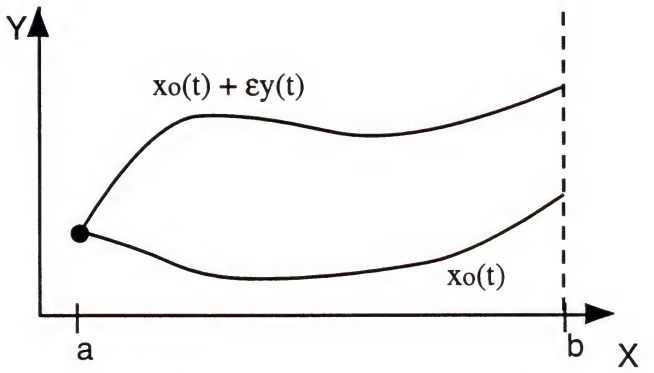


Figure 2.4 Variations of a Solution with Incomplete Boundary Conditions

2.5 Equivalence of Using Energy and Length Functions in Path Minimization

The last task in this chapter is to show the equivalence between minimizing the integral equation of the length and the integral equation of the energy function. In a standard optimization problem, the minimizing of two objective functions over a finite set

of variables is equivalent only if they both yield the same solution vector. The analogous situation in the calculus of variations is that the minimization of the length and energy functions are equivalent if their solution curves are identical. To show this for calculus of variations problems it is sufficient to show that the Euler-Lagrange equation yield the same second order differential equations in both cases. This will be shown for certain illustrative cases. In order to show this for the general case it would be necessary to use tensor calculus and notation, which is beyond the scope of this research. A general proof is given in Chapter 9 of A Comprehensive Introduction to Differential Geometry [Spiv 79].

In order to determine for which cases the equivalence of minimizing the length and energy functions will be shown, it is necessary to gain an insight into the form that the solution curve will take. Shcherbak gives a set of criteria for the solution curve around a single obstacle in Euclidean space in his paper "Wavefronts and Reflection Groups" [Shc 88]. He describes the solution as a piece wise smooth curve consisting of three geodesic segments. The first is a straight line segment from the start position to a point on the surface of the obstacle. The point on the obstacle is chosen such that the line segment is tangent to its surface. The second segment is a geodesic on the surface of the obstacle. The final curve is a straight line segment from a point of tangency on the obstacle to the end point. This can then be generalized to an arbitrary number of obstacles by stating that the piece wise smooth solution curve will be made up of a series of geodesics, either through Euclidean space or along the surface of an obstacle. In this research the obstacles are spherical or elliptical, it will therefore be sufficient to show that the length and energy functions provide the same critical solutions for geodesics in Euclidean space and on the surface of a sphere. The cases of circular or elliptical obstacles in the plane and ellipsoidal obstacles in three dimensional space will be considered modifications of geodesics on a sphere.

Equivalence of the energy and length functions can be shown algebraically by deriving the two sets of differential equations using the Euler-Lagrange equation and then showing that the two systems of second order differential equations are equivalent. The details of these calculations are summarized below. Briefly, the Euler-Lagrange equation

of a length function has a radical term in the denominator. This can be eliminated algebraically, in turn producing the same set of differential equations as those generated by the Euler-Lagrange equations of the energy function. It should be pointed out that this method is a brute force approach and that more elegant methods exist in differential geometry. For the sections of the path in in ambient space the length or energy functions are unconstrained and the Euler-Lagrange equations for both the length and energy give a set of differential equations in the form

$$\begin{Bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \vdots \\ \ddot{x}_n \end{Bmatrix} = 0 \qquad (2.21).$$

The solution to this system of equations is a vector of linear equations in the parameter t. This gives a line in the N dimensional space which is intuitively recognized as the geodesic in Euclidean space.

To derive the equation for a geodesic on the surface of a sphere requires it is necessary to use spherical coordinates. The square of an infinitesimal element is given by

$$ds^2 = d\rho^2 + \rho^2 d\theta^2 + \rho^2 \sin^2\theta d\varphi^2 \qquad (2.22).$$

If equation (2.14) is converted to spherical coordinates it can be observed that an active constraint is equivalent to setting ρ equal to a constant, the equation under the integral for a sphere centered on the origin becomes

$$E_a^b = \int_a^b [\dot{\theta}^2 + \sin^2\theta \dot{\phi}^2]dt \qquad (2.23).$$

The stationary curves for both the energy integral and the the length integral are given by the differential equations

$$\ddot{\theta} - \dot{\phi}^2 \sin\theta \cos\theta = 0 \qquad (2.24a),$$

$$\ddot{\phi} \sin^2\theta + 2\,\dot{\phi}\,\dot{\theta} \sin\theta \cos\theta = 0 \qquad (2.24b).$$

The solution of this system of equations are the arcs of great circles on a sphere.

As a final step in the discussion of the optimal solutions of path planning problem it is necessary to introduce the Principle of Optimality. The Principle of Optimality states that if a given curve minimizes an integral over the interval [a,b] it must minimize the integral over the interval [a,c] where $a < c < b$. This can be stated more clearly for the path planning problem in the following manner. If a given path is a minimum length path from the start to the goal position, it must also contain the minimum length path from the start position to any intermediate point on the path. In Figure 2.4 a piece wise smooth curve is given, in which all the segments are geodesics, but violates the Principle of Optimality. The shortest distance from the start position to any point in segment three clearly is not contained in the path shown. Therefore this path could not be obtained from critical solutions of the Euler-Lagrange equation.

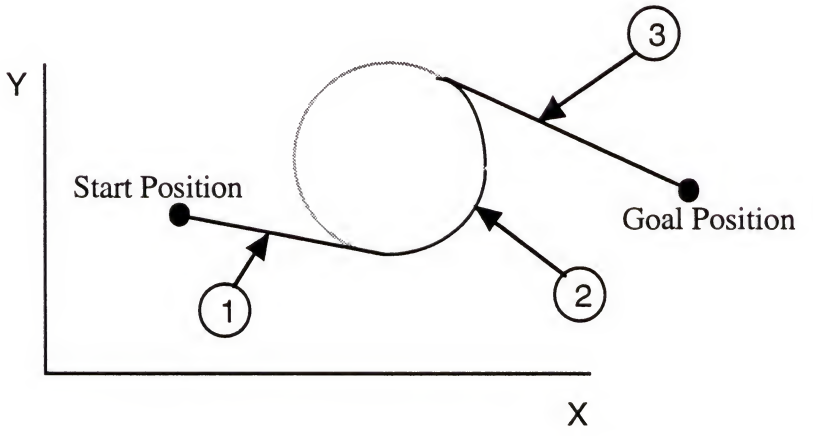


Figure 2.5 A Non Optimal Path Comprised of Geodesic Segments

CHAPTER 3
COMPUTATIONAL ALGORITHM FOR THE GENERAL PATH PLANNING PROBLEM IN EUCLIDEAN SPACE

3.1 Methods for Approximating Stationary Curves

It has been established that the path planning problem can be formulated as an unconstrained minimization and that the Euler-Lagrange equation can be used to analytically obtain solutions to this problem. However, analytical solutions are limited in their scope and the task now remains to construct a suitable computer algorithm from the theory developed in the last chapter. New methods for numerically determining the stationary curves of an integral equation have been developed by Gregory and Lin [Greg 92] and they form the basis of the algorithms used in the rest of this dissertation. For a general calculus of variations problem their techniques vary depending on whether the problem involves boundary conditions or initial value conditions, but the outline given in this section will focus on developing an algorithm to solve problems with partial boundary and transversality conditions. For convenience a restatement of the path planning problem is given below;

$$\text{Minimize } I(x) = \int_0^1 F[t,X(t),\dot{X}(t)]dt \tag{3.1a},$$

where the function under the integral is given by

$$F(t,X,\dot{X}) = \sum_{i=1}^{N} \dot{x}_i^2 - \sum_{j=1}^{M} \dot{x}_{2j-1+N}\{\sum_{k=1}^{N} (x_k - a_{jk})^2 - r_j^2 - \dot{x}_{2j+N}^2\} \tag{3.1b},$$

for N dimensions and M obstacles, and the boundary conditions by

$$X(0) = \begin{pmatrix} x_1 start \\ \vdots \\ x_N start \\ 0 \\ \vdots \\ 0 \end{pmatrix} \qquad (3.1c),$$

and

$$x_1(1) = x_1 goal$$
$$\vdots$$
$$x_N(1) = x_N goal \qquad (3.1d).$$

The most important feature of Gregory and Lin's algorithm is that they use the first variation of the functional being minimized as the basis of their approximation. This approach closely follows the derivation of the Euler-Lagrange equation and is a departure from other techniques based on variational principles. For instance, Pontryagrin's Minimum Principle used in optimal control is derived from the Hamiltonian function and can only be solved for simple problems, and dynamic programming based on Bellman's Optimality Principle requires a descretization of the configuration space and therefore falls into the category of the decomposition methods described in Chapter 1. The first variation I' (x,y) of equation (3.1a) is restated here as

$$I'(x,y) = \int_0^1 [F_x y + F_{\dot{x}} \dot{y}]dt = 0 \qquad (3.2),$$

where the solution vector is given by $x_o = (x^1(t),x^2(t),...,x^{N+2M}(t))$ and the vector of arbitrary variations is given by $y = (y^1(t),y^2(t),...,y^m(t))$. The variation I' (x,y) is the first order term in the Taylor series expansion of the objective function (3.1a) and setting this to zero is equivalent to setting the first derivative of the objective function to zero in a standard

optimization problem. It is important to reiterate that equation (3.2) gives a necessary condition for the minimization of the energy function and that an approximation of a solution vector $x_o(t)$ which satisfies this integral is sought.

The interval of integration for the first variation is divided into a series of equal sub-intervals where the step size h is given by (b-a)/P. The integral is then approximated by an equation over each interval $[a_{k-1}, a_{k+1}]$ where k ranges from [1, P]. It is important to note that if a solution curve $x_o(t)$ has been determined the value of the first variation over any sub interval must also be zero. This follows from the observation that the solution curve over $[a_{k-1}, a_{k+1}]$ is embedded in the solution over the whole interval [0,1] as shown in Figure 3.1. If $x_o(t)$ is the optimal curve from [0,1], then it must also be the optimal curve over $[a_{k-1},a_{k+1}]$ and the first variation over this sub interval must also be zero.

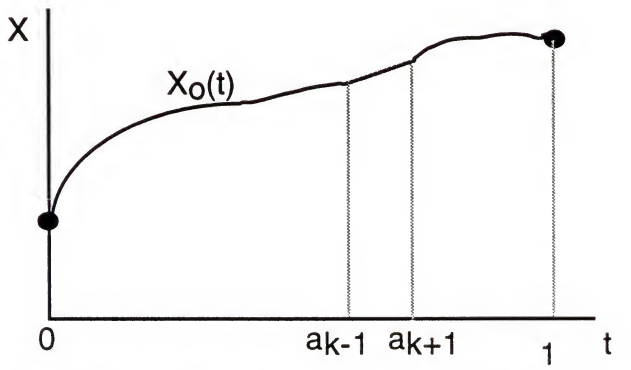


Figure 3.1 Embedding of the Solution Curve over a Sub-Interval

It can be observed that each sub interval overlaps with adjacent sub intervals. This causes a coupling in the finite difference equations and thus forces an interdependence between the solution curves. The next step is to approximate the functions under the integral. In the derivation of the Euler-Lagrange equation in the previous chapter it was stated that the variational curves $y = (y^1(t),y^2(t),...,y^m(t))$, are completely arbitrary. The variation can

then be represented by linear interpolation functions. The derivatives of the objective functions $\partial f/\partial x$ and $\partial f/\partial \dot{x}$ are approximated by finite difference equations which are calculated explicitly from equation (3.1b). The first variation can now be approximated over all sub-intervals except for the right endpoint which has to account for the partial boundary conditions by

$$\begin{aligned} I'(x,y)_{[a_{k-1},a_{k+1}]} = \\ G_k &= f_{\dot{x}}(a^*_{k-1}, \frac{x_k+x_{k-1}}{2}, \frac{x_k-x_{k-1}}{h}) + \frac{h}{2} f_x(a^*_{k-1}, \frac{x_k+x_{k-1}}{2}, \frac{x_k-x_{k-1}}{h}) \\ &\quad -f_{\dot{x}}(a^*_k, \frac{x_{k+1}+x_k}{2}, \frac{x_{k+1}-x_k}{h}) + \frac{h}{2} f_x(a^*_k, \frac{x_{k+1}+x_k}{2}, \frac{x_{k+1}-x_k}{h}) = 0 \end{aligned} \tag{3.3},$$

where $k = 1,2,...P-1$, and

$$a^*_k = \frac{a_k + a_{k+1}}{2} \tag{3.4}.$$

Each G_k is a vector equation of length N+2M, where N is the number of dimensions and M is the number of obstacles. The last equation has to take into account the boundary conditions for the physical variables and the transversality conditions for the Lagrange and slack functions. These are given without proof by Gregory and Lin [Greg 92] as

$$G_P = (\bar{0}, \overline{G}_P) \tag{3.5},$$

where the transversality conditions are approximated by

$$\overline{G}_P = f_{\dot{x}}(a^*_{P-1}, \frac{x_P+x_{P-1}}{2}, \frac{x_P-x_{P-1}}{h}) + \frac{h}{2} f_x(a^*_{P-1}, \frac{x_P+x_{P-1}}{2}, \frac{x_P-x_{P-1}}{h}) = 0 \tag{3.6}.$$

The equations can now be combined into a single vector

$$G = \begin{pmatrix} G_1 \\ \vdots \\ G_P \end{pmatrix} = (0) \tag{3.7}.$$

The system G contains a total of P*(N+2M) equations. The minimization problem has now been converted into one of finding the roots x_k of a system of nonlinear equations. The obstacles used in this research are convex parametric curves and surfaces whose derivatives can be calculated explicitly. The preferred method for solving this problem is the Newton-Raphson Method. The numerical scheme for this method is given in equations (3.8) and (3.9) where the roots X of G(X) can be approximated using iteration. These equations are given as follows:

$$\Delta X = -G(X^{(n)}) * \left[\frac{\partial G(X^{(n)})}{\partial X}\right]^{-1} \tag{3.8}$$

$$X^{(n+1)} = X^{(n)} + \Delta X^{(n)} \tag{3.9},$$

where n represents the nth iteration. This is done until X converges on a solution. The matrix of derivatives is block tri-diagonal and is given by

$$\left[\frac{\partial G(X^{(n)})}{\partial X}\right] = \begin{bmatrix} \frac{\partial G_1}{\partial X_1} & \frac{\partial G_1}{\partial X_2} & & & & \\ \frac{\partial G_2}{\partial X_1} & \frac{\partial G_2}{\partial X_2} & \frac{\partial G_2}{\partial X_3} & & & \\ & & \ddots & & & \\ & \frac{\partial G_k}{\partial X_{k-1}} & \frac{\partial G_k}{\partial X_k} & \frac{\partial G_k}{\partial X_{k+1}} & & \\ & & \ddots & & & \\ & & & \frac{\partial G_P}{\partial X_{P-1}} & \frac{\partial G_P}{\partial X_P} \end{bmatrix} \tag{3.10}.$$

Each square sub matrix has dimension (M+2N) and the two matrices in the last row are given by

$$\frac{\partial G_P}{\partial G_{P-1}} = \begin{bmatrix} 0 \cdots 0 \\ \\ \dfrac{\partial \overline{G}_P}{\partial X_{P-1}} \end{bmatrix} \tag{3.11}$$

and

$$\frac{\partial G_P}{\partial G_P} = \begin{bmatrix} 1 \cdots 0 \\ \\ \dfrac{\partial \overline{G}_P}{\partial X_P} \end{bmatrix} \tag{3.12},$$

to account for the transversality conditions.

3.2 Computational Algorithm

The key to root finding for a nonlinear systems of equations is to have a good initial guess X^0 that is in the neighborhood of convergence for a given root. This has several important implications for root finding in general and for path planning in particular. Firstly, there is the question of how many roots are there to the system of equations $\{G\}$, given by equation (3.6) and if there is more than one root, which ones will generate "efficient" paths? Secondly, will an easily generated initial guess or more specifically a guess that requires very few computations, be in the neighborhood of convergence of an efficient root? Answers to these questions will be drawn from the experimental results given in Chapters 4 and 5. At the present time the initial guess will be kept sufficiently general so that it can be modified to test the convergence of a variety of problems.

Once a simple rectilinear path is specified as the initial guess for the physical variables, it is normalized as a function of the parameter t and its value at each partition a_k on the interval is determined. The slack functions can then be calculated from these functions. The derivatives of the slack function are a measure of the distance from the point on the path at each a_k to the boundary of each obstacle. In the planar case, the slack derivative for each obstacle is calculated by

$$\dot{x}_s^2 = \{(x_1 - a_1)^2 + (x_2 - b_1)^2 - r_1^2\} \qquad (3.13),$$

and the slack function is approximated by integration where the left hand boundary condition is set to zero by equation (3.1c).

For an initial guess where the physical path does not contact any obstacles, the derivatives of the Lagrange function for each obstacle should, theoretically be equal to zero. This, however, leads to a practical problem in the numerical solution. If the derivatives of the Lagrange functions are set to zero in the initial guess, the objective function reduces to

$$\int_0^1 \sum_{i=1}^{N} \dot{x}_i^2 \, dt \qquad (3.14),$$

and the Lagrange functions remain zero throughout all iterations. Computationally the problem reduces to finding the shortest distance between the start and end points while ignoring the obstacles. In order to correct this problem, the Lagrange functions are set to range from 0 to 1, giving a nominal value for the derivative to start the calculations.

The other practical concerns of interest are the solution of the matrix system of equations (3.7) and the criteria for convergence. The matrix system of equations is solved using LU decomposition with back substitution as described in Numerical Recipes in C, The Art of Scientific Computing [Pres 88]. This method is a robust, general algorithm for

solving matrix equations and takes $O(^1/_3n^3)$ computations, where n is the dimension of the matrix. It does not take into account any special properties of the matrix being inverted, whereas the matrix $[\partial G]$ from equation (3.12) has several properties that could be exploited to gain some improvements in speed. A discussion of the possible improvements is postponed until Chapter 6. The convergence of the Newton-Raphson method is checked by summing the squares of ΔX and checking it against a suitably small epsilon, on the order of 10^{-8}. The algorithm for the path planning problem is now presented in Figure 3.2.

3.3 Expected Form of the Solution

In this section some observations will be made about the expected form of the solutions to the path planning problem and these observation will then be compared against actual results. It is useful to examine the solutions for simple cases in order to come to an overall understanding of the behavior of the algorithm. The concept of a minimum length path is fairly intuitive, but the Lagrange and slack functions are understood in terms of their derivatives and require more description. It is the derivative of the Lagrange function which multiplies the obstacle constraint in equation (3.1b) and it is the square of the derivative of the slack function which transforms the inequality constraint into an equality constraint. In order to simplify the description of the predicted solution, a planar example will be considered with one obstacle as shown in Figure 3.3.

The physical variables of a solution curve $x_o(t)$ to the problem of bypassing an obstacle are expected to navigate a minimum length path around the obstacle and at certain points to come in contact with its boundary. The derivative of the Lagrange function is equivalent to a Lagrange multiplier in standard optimization theory and should be non zero when the constraint is active. Thus, over the same path the derivatives of the Lagrange function will be zero before the obstacle is contacted, non zero when the path contacts the obstacle, and zero again when the path is no longer in contact with the obstacle. With the left hand boundary condition given in equation (3.1c) the derivatives can be integrated to

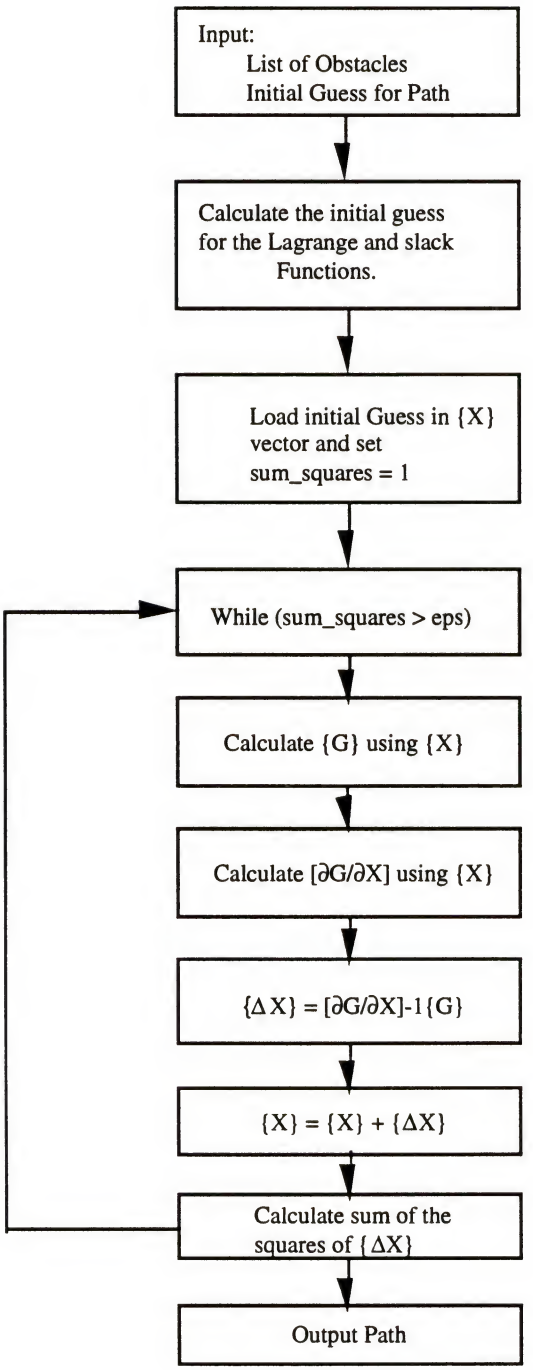


Figure 3.2 Path Planning Algorithm Using Calculus of Variations

obtain a unique curve. The behavior of the slack function can be described in a similar manner. The slack function's derivative at a given time t is equal to the shortest distance between a point on the path at t and the boundary of the obstacle. Therefore the derivatives along the path will start out at some positive value, decreasing until the obstacle is contacted, be zero valued while the path travels along the obstacle, and finally increase as the path moves away from the obstacle. Because the squares of the slack function are used in the formulation of the problem, both positive and negative values can satisfy the constraint. However, since the positive values are chosen in the initial guess, for this description, it is assumed that these values are in the neighborhood of a root with similar positive values. Finally the switching behavior of the Lagrange and slack derivatives can be seen over the path in a similar manner to standard optimization theory. Namely, the derivatives of one of the slack or Lagrange functions is zero when the derivatives of the other function are non zero.

This idealized solution curve will now be compared with the solutions obtained in two examples. In the first case an asymmetric path is sought around the circular obstacle shown in Figure 3.3. Two initial guesses are given, the first on the same side of the obstacle as the solution path and the second on the opposite side of the obstacle. This shows the ability of the algorithm to pass through an obstacle to reach a solution path. Each initial guess is tested for three step sizes and the results of the convergence to a solution are shown in Table 3.1. It can be seen from the table that it is relatively fast and that a unique solution is given with path lengths all approximately the same. The paths of the solution are plotted in Figure 3.3 with the obstacle to show that an efficient path is generated which bypasses the obstacle as predicted in the discussion above. The following two graphs plot the Lagrange and slack functions respectively for the last test case in Table 3.1 with initial guess number 2 and a path with fifteen steps. The Lagrangian function behaves as expected. Its derivative is zero valued when the path is not in contact with the obstacle, non zero when the path is in contact with the obstacle, and zero once more when the path leaves the obstacle boundary.

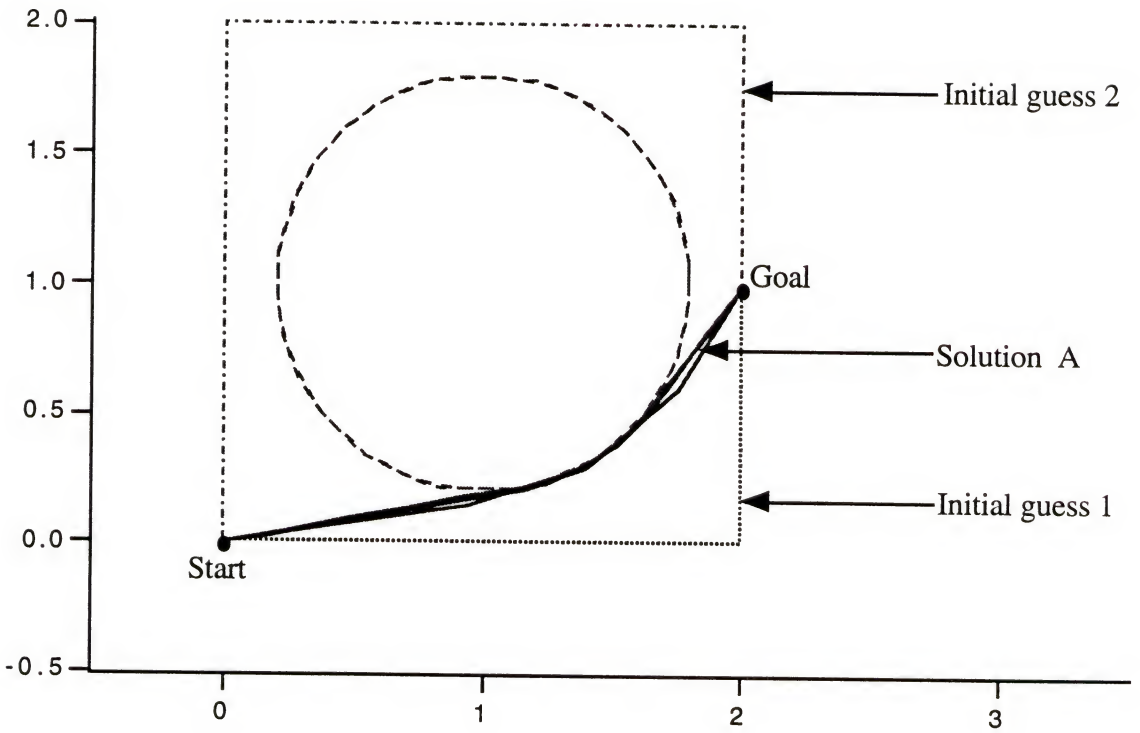


Figure 3.3 Case 1 - Bypassing a Circular Obstacle

Table 3.1 Convergence Data for Case 1

Initial guess	Number of steps	Size of matrix	Convergence	Number of iterations	Path length
1	5	20	soln A	14	2.372
1	10	40	soln A	17	2.365
1	15	60	soln A	27	2.362
2	5	20	soln A	15	2.372
2	10	40	soln A	20	2.364
2	15	60	soln A	18	2.362

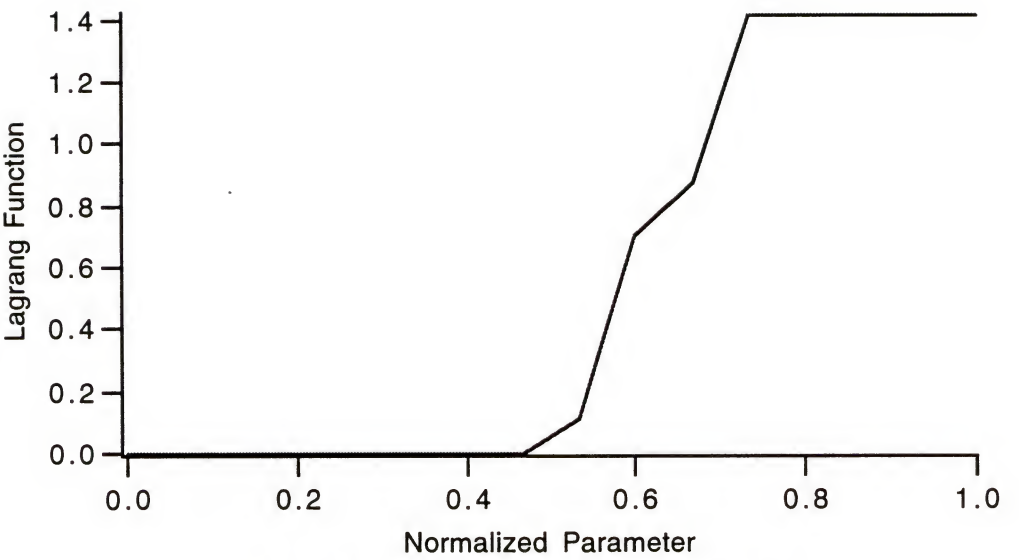


Figure 3.4 The Lagrange function for Case 1

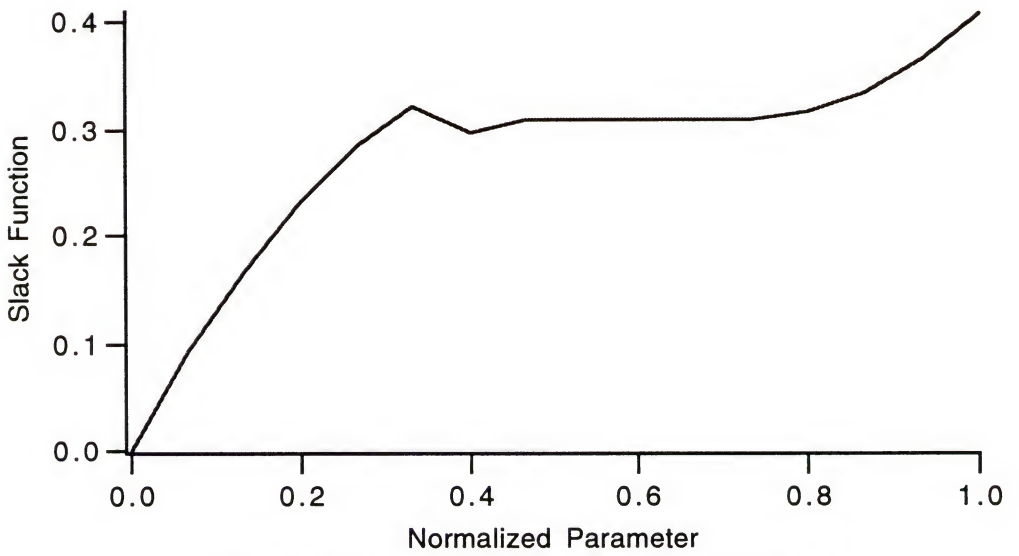


Figure 3.5 Normalized Slack Function for Case 1

The slack function also behaves in the desired manner. The derivative decreases as the obstacle is approached, settles to a value of zero while in contact with the obstacle, and increases once the path leaves the obstacle. The values of the derivative are all greater than zero over their range and the function has converged to a root in the neighborhood of its initial guess.

In the second test case the same circular obstacle is used with the goal position adjusted to introduce symmetry into the problem. Symmetry introduces further challenges into solving the problem which points out a significant difference between the analytical equations and the numerical algorithm. In Figure 3.6 several solutions are shown which pass on either side of the obstacle. In terms of the Euler-Lagrange equation, both solutions would satisfy the same set of second order differential equations and only differ in their constants of integration. In the numerical algorithm these constants of integration are implicitly chosen and cause difficulties in convergence for symmetric cases where two equally valid solutions exist. This can be seen in comparing the number of iterations needed to converge in Tables 3.1 with the number of iterations needed to converge in Table 3.2. This problem with convergence can also be seen in the final graphs of the Lagrange and slack functions.

There are once again three step sizes used for each initial guess. In general the number of iterations for convergence is a lot larger for the symmetrical cases and the quality of the solution paths is not as good as the paths in the previous case. It should be noted that for one of the steps sizes the solution jumps from the nearest solution to the solution path on the other side of the obstacle.

The Lagrange and slack functions for the last case in Table 3.2 consisting of fifteen steps and converging to solution B are shown below. It is straight forward to see that these solution curves are not smooth functions and that in general there is more difficulty in convergence. It can also be seen that certain derivatives for the slack function are now negative showing that the problem has converged to a different root. What is still evident is the switching behavior in the two sets of derivatives.

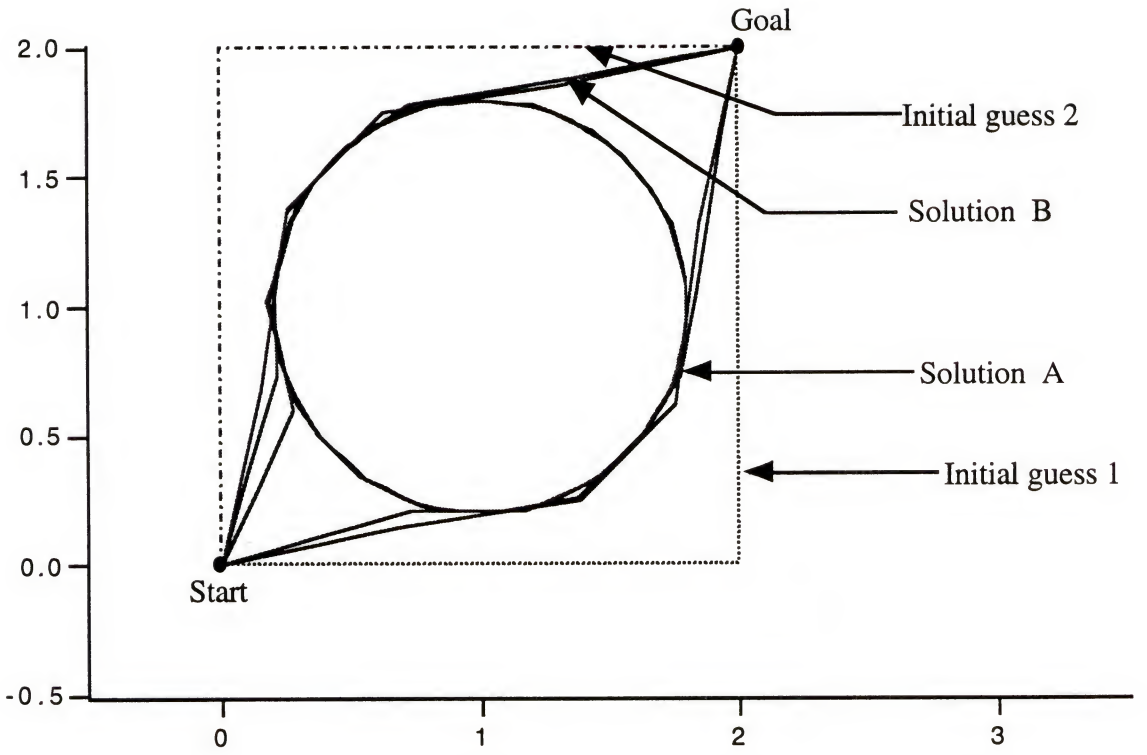


Figure 3.6 Case 2 - Bypassing a Circular Obstacle With Symmetry

Table 3.2 Convergence Data for Case 2

Initial guess	Number of steps	Size of matrix	Convergence	Number of iterations	Path length
1	5	20	soln A	44	3.331
1	10	40	soln A	56	3.32
1	15	60	soln B	138	3.354
2	5	20	soln B	44	3.331
2	10	40	soln B	58	3.32
2	15	60	soln B	114	3.355

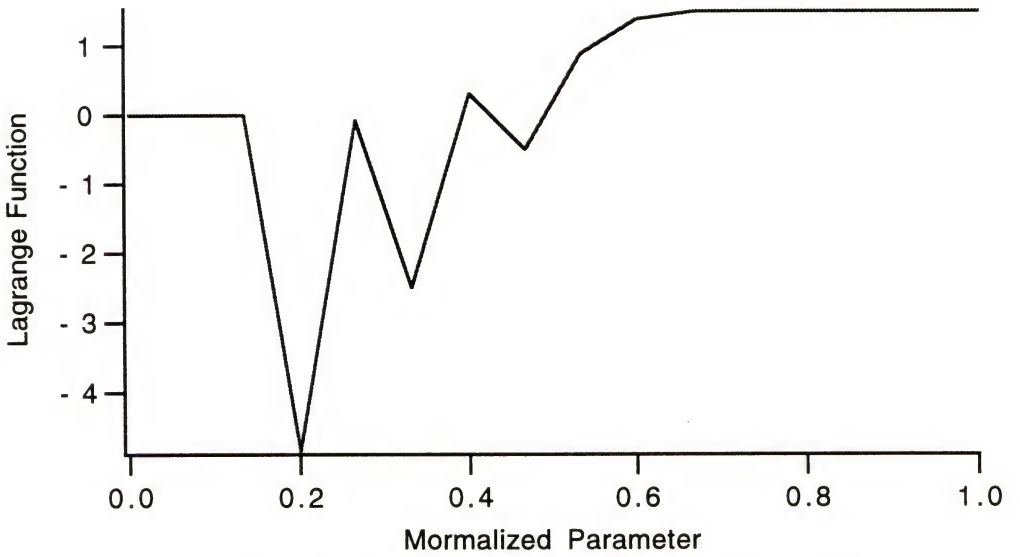


Figure 3.7 Lagrange Function for a Symmetric Case

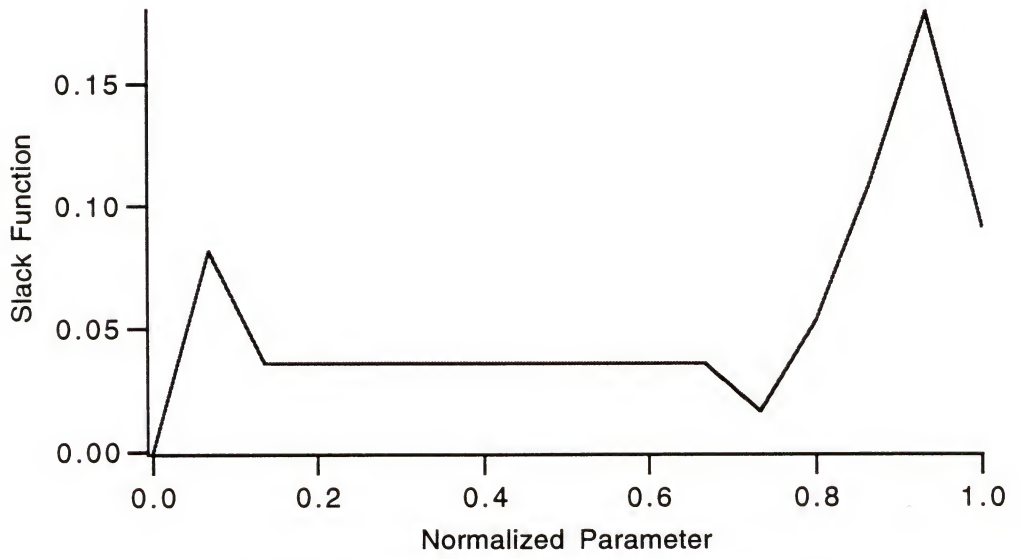


Figure 3.8 Slack Function for a Symmetric Case

3.4 Modifications to the Algorithm

The examples from the previous section give a overall view of the form of the solution curves for two different cases. It will be shown in the next Chapter that the non smooth behavior of the slack function occurs in many problems where the algorithm has not converged to a good solution and these functions can be used as a qualitative test to judge the path that has been found by the algorithm. At the present time the information from the previous section will be studied to see if any modifications can be made to the present algorithm.

A second solution curve for the slack function in Figure 3.8 can easily be constructed by forcing all the derivatives to be positive thus generating a curve similar to the one in Figure 3.5. This can also be done algorithmically by performing a check on the derivative after each iteration and replacing any negative derivatives with its absolute value. This has be tested for several examples and unfortunately caused an instability in the algorithm, and a divergence from the initial guess.

When the problems for the symmetric case with one obstacle are put in perspective with the test cases in the next two chapters, it can be seen that the solution is still acceptable and leads to an efficient path. This suggest a modification to the algorithm that can be employed when the convergence with many obstacles is virtually impossible. A stage wise optimization can be performed by adding one obstacle at a time. This algorithm is shown in Figure 3.9 and works by starting off with one obstacle and reaching a solution, then adding another obstacle and using the solution path from the previous step as the initial guess in the current step. This process is repeated until all the obstacles have been added and a path has been found. The idea for this modification has come from stage wise optimization but some important differences should be noted. Firstly unlike stage wise optimization, the modified algorithm is not segmenting the path and optimizing each section. Secondly, the path is determined in the final step where the refined initial guess, over the whole path, is used to determine a final solution.

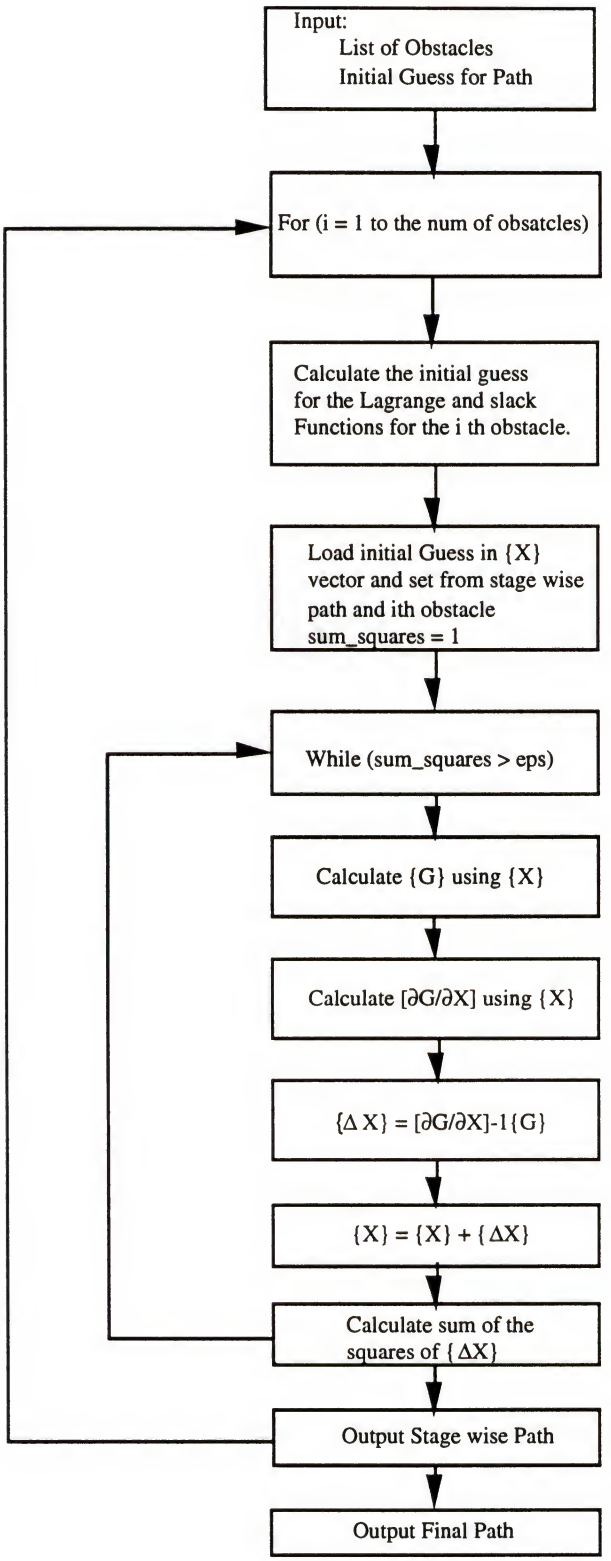


Figure 3.9 Stage Wise Optimization in the Path Planning Algorithm

CHAPTER 4
PATH PLANNING IN THE EUCLIDEAN PLANE

4.1 Selection of Test Cases

The purpose of this chapter and the next is to present test cases that demonstrate the strengths and weaknesses of the algorithm that has been developed. The first task will be to generate timing curves for the solution of a path planning problem in the plane. Most of the computation time is consumed solving for the system of matrix equations given by equation (3.8). This is done using the LU decomposition algorithm for inverting a matrix which has a fixed time of $O(1/3n^3)$. The size n of the matrix is calculated using the formula n = (N + 2*M)*P, where N is the number of dimensions, M is the number of obstacles, and P is the number of steps in the path. The matrix equation is solved in the iteration loop, which complicates the timing of the algorithm. However, it is expected that the overall computational time will rise with an increase in complexity of the path planning problem. The timing tests are presented in Section 4.2.

In the previous chapter the observations were made regarding the form of the slack functions for simple examples. For cases in which the algorithm finds the desired root it was observed that these functions are smooth and close to the initial guess that has been chosen. It was further shown that the functions become less well behaved when the algorithm has difficulty converging to a solution. In Section 4.3 further tests will be done on bypassing a single obstacle to show the different problems that can arise. In these cases the slack functions will be examined to check the form of the solution. As was described in Section 3.4 these functions can be used to judge the quality of a solution, but cannot be used to manipulate convergence to a desired root.

A second issue which needs to be addressed is the introduction of instabilities as the complexity of the obstacle geometry increases. The effects of these instabilities will be seen on the physical functions which describe the actual path. These instabilities appear by generating different paths under slightly varying conditions. Some of these paths are valid and some violate obstacle boundaries leading to invalid paths. It is felt that these instabilities are caused by the numerical methods employed and not by the theoretical derivation of the path planning problem. In the study of the numerical solution of differential equations these instabilities appear in the form of parasitic roots which are generated by round off errors. This issue will be discussed further in Chapter 6. At the present time the focus will be on determining when an invalid solution has been generated, and methods for dealing with this problem. In Section 4.4 general obstacle geometries will be considered and the focus will be on identifying the invalid roots. When there is a difficulty with converging to a good root, some success has been realized by switching to the stage wise optimization and this will be demonstrated in Section 4.5.

Finally, in Section 4.6 moving obstacles are considered. It is shown that, while theoretically, the functions under the integral can be functions of time, in a practical sense these problems are too unstable for computer solution and there is a problem with convergence to a meaningful solution in even the simplest problem.

4.2 Timing Test

It is expected that the CPU time required to find a solution will increase with the number of obstacles. There is a tendency for the number of iterations to vary for different configurations involving a given number of obstacles. The geometry has been fixed into a rectangular array in an attempt to reduce these effects. The configuration of the array is shown in Figure 4.1 for twelve circular obstacles. The CPU time ranges from a couple of seconds to half an hour, roughly increasing with the complexity of the problem. For each array initial guesses using an upper and lower path have been tested with the number of steps ranging from ten to thirty in increments of five. This has been done to

show that the convergence to a solution is independent of the initial guess in these cases. The stray solution shown in Figure 4.1 is for a step size of ten which is coarse enough to cause problems with convergence. This problem disappears as the number of steps is increased. In general the algorithm produces efficient paths and as the number of steps increase the path more easily goes around obstacles instead of cutting through the edges.

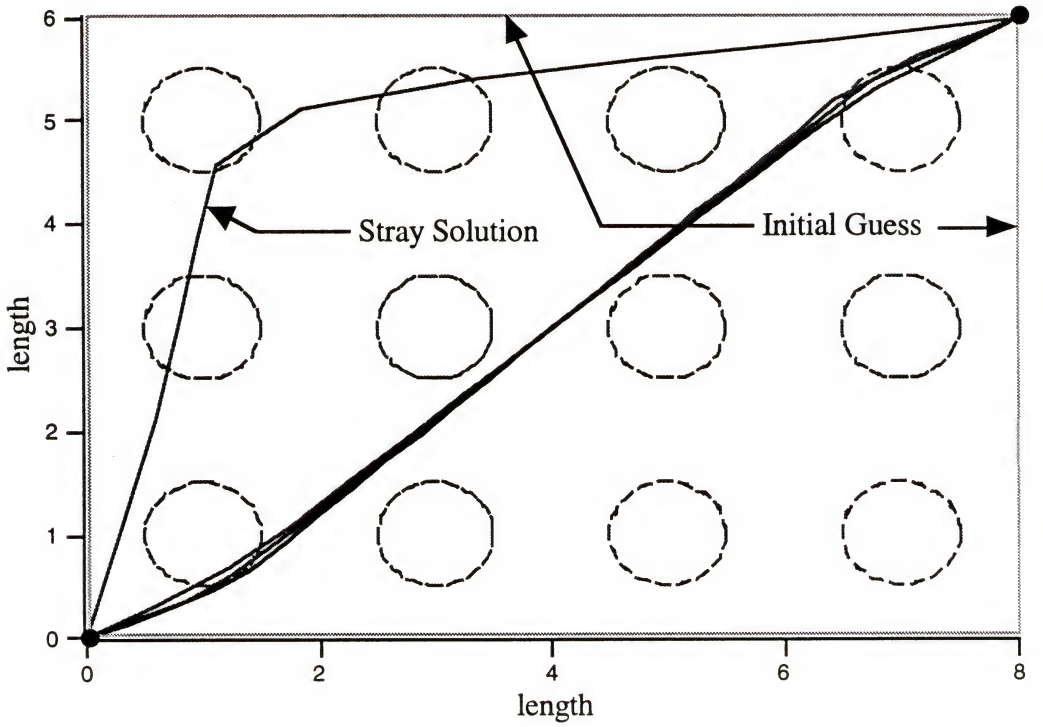


Figure 4.1 Timing Test Configuration for Twelve Obstacles

The number of iterations versus the matrix size is shown in Figure 4.2. The high point in the left end of the curve for twelve obstacles corresponds to the stray path shown in Figure 4.1. For the most part the number of iterations falls within a bounded region. In Figure 4.3 the CPU time in seconds is plotted against the size of the matrix array. The tests were run on a Silicon Graphics VGX with a MIPS R4000 CPU running at 100 MHz. A general trend of an increase in CPU time with an increase in matrix size can be seen.

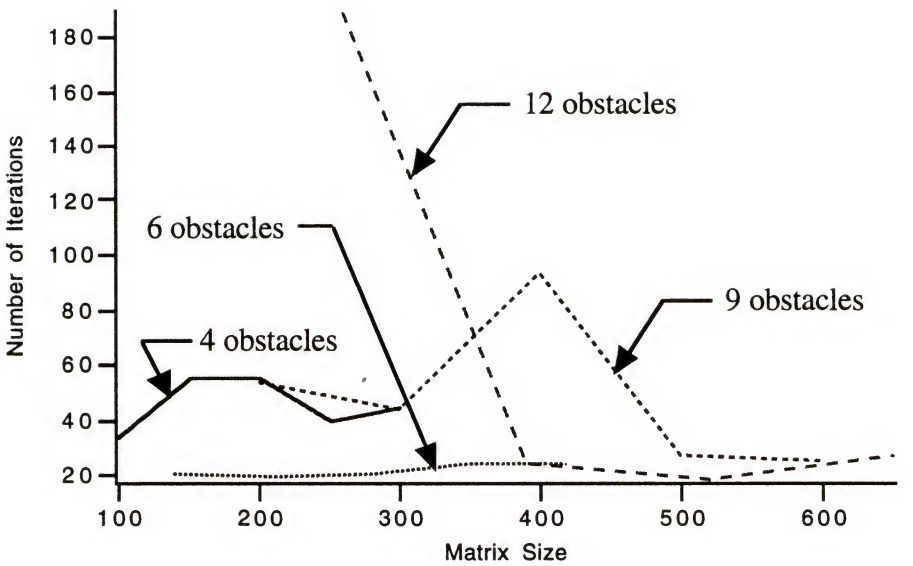


Figure 4.2 Number of Iterations vs Matrix Size

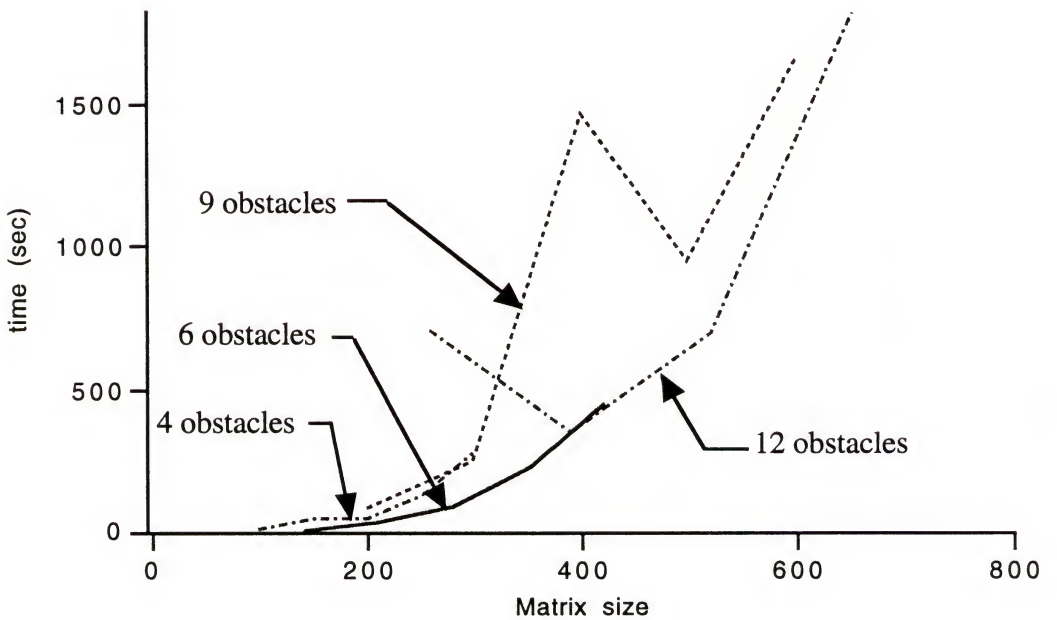


Figure 4.3 CPU Time vs Matrix Size

4.3 Further Study of Path Planning Around a Single Obstacle

In Chapter 3, two simple examples were presented for bypassing a circle in the plane. This was done to introduce the form of the slack and Lagrange functions for a solution and to show how these functions degenerate when the algorithm has difficulty in finding a solution. In this section further study is done on a single obstacle in the plane, where the effects of deforming a circle into an ellipse are studied. In the first example, shown in Figure 4.4, with corresponding slack functions in Figure 4.5, a symmetrical case is studied where there are two shortest length paths. In Figure 4.4, Case (a), the obstacle is a circle with a radius of 0.5. The radius on the X axis is increased in successive steps to where in Case (d) it is 2.0. For a radius of 2.5 the algorithm will not converge. As the obstacle is deformed into an ellipse the algorithm is able to find a valid path up until Case (d), where the solution path has degenerated into an instability. The problem is ill conditioned and the algorithm finds a parasitic root. As can be seen by examining the slack functions, none of the paths would pass the slack function test whereas Cases (a),(b) and (c) generate valid paths. In Figure 4.6 the same ellipse is studied with a dimension along the X axis of 2.5. The purpose of this test is to come to get an indication of the sensitivity of the algorithm to symmetry. In this test the ordinate of the start and end points approaches 3.0 which is the point of symmetry. In Case (a) the value of the ordinates is 3.1 and in Case (b) the value of the ordinates is 3.01. When the ordinate decreases to 3.001, the problem diverges and no solution is found. The corresponding slack functions, shown in Figure 4.7, are smooth for both cases.

In the next set of tests the effects of an increasing curvature is shown on the algorithm. In Figure 4.8 the start and end points of the path are moved to eliminate the effects of symmetry and the obstacle is deformed from a circle into an ellipse. As can be seen in Figure 4.8 as the curvature increases the solution path starts to wrap around the obstacle. The slack functions corresponding to these cases are shown in Figure 4.9. As can be seen the slack functions are smooth and behave as expected. As the sharpness of the curvature increases to the level shown in Case (d) the path generated is less than optimal. However, using the slack function test the path is considered valid.

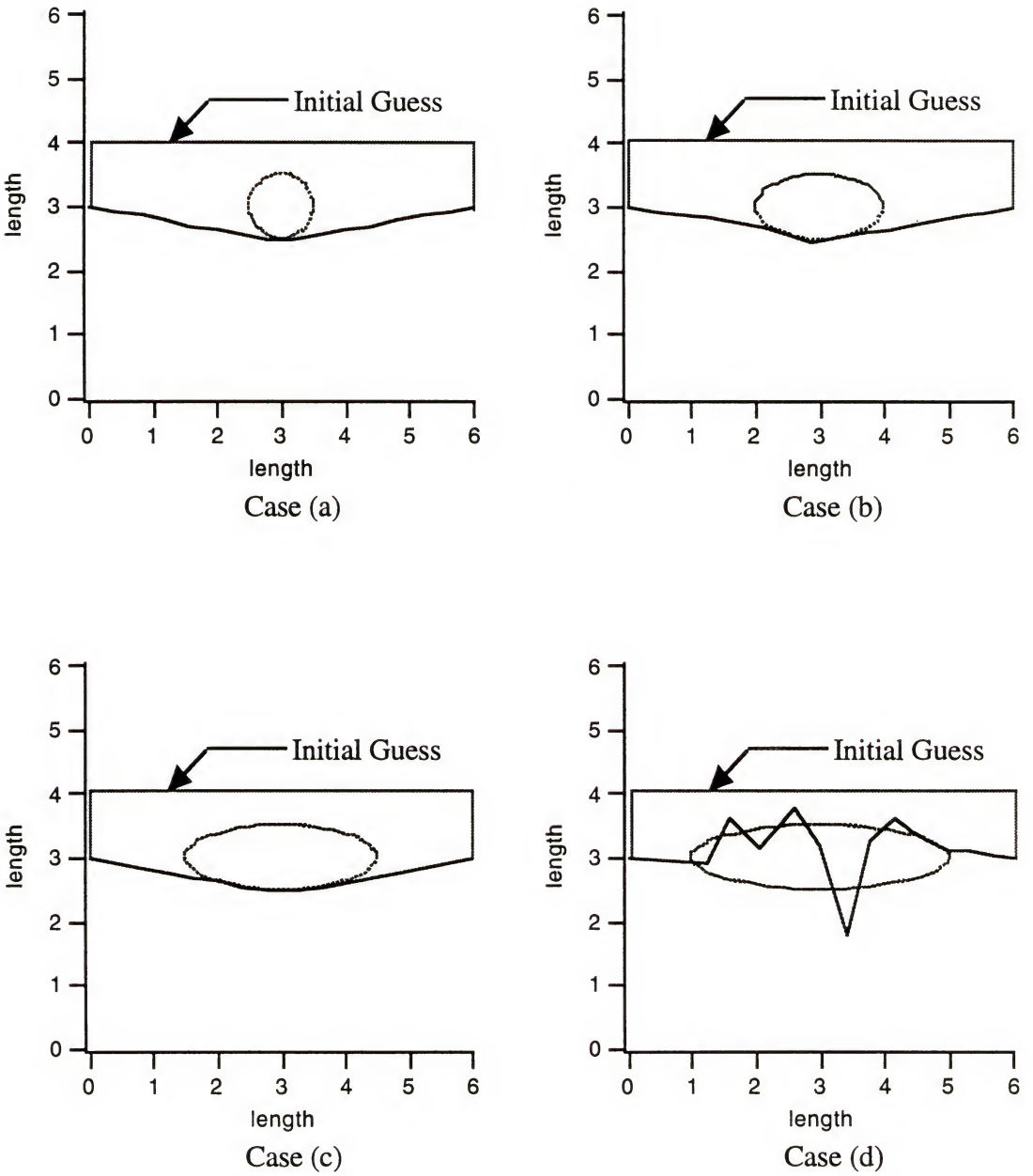


Figure 4.4 Study of the Paths Generated While Deforming a Circle into an Ellipse

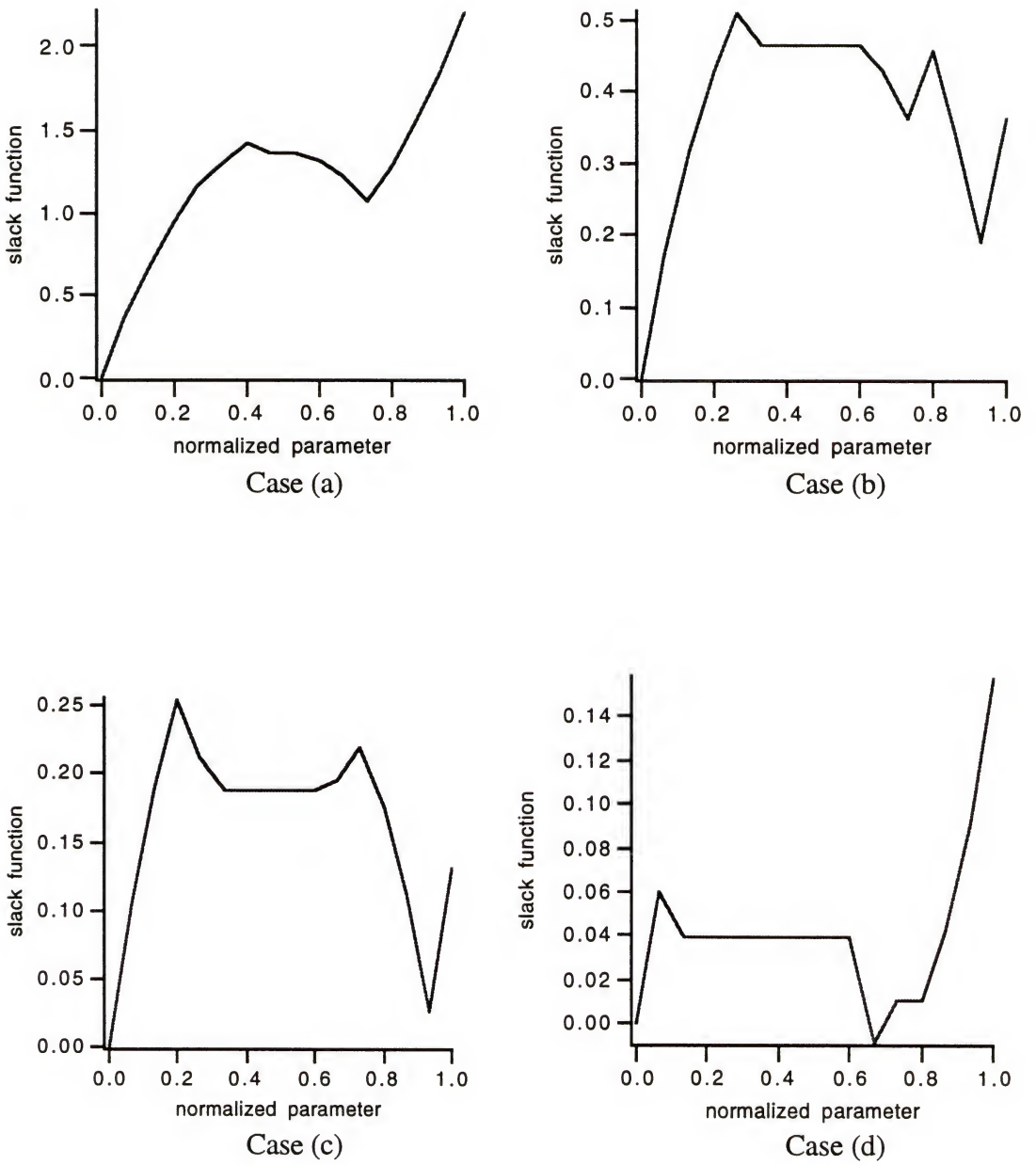


Figure 4.5 Slack Functions for the Deformation of a Circular Obstacle into an Ellipse

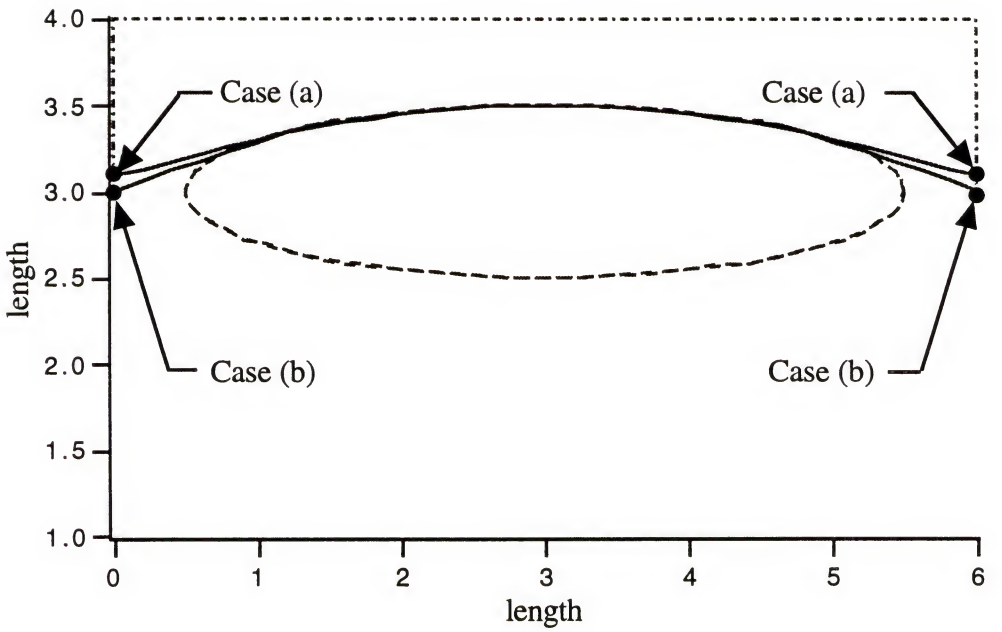


Figure 4.6 Paths Approaching Symmetry Around an Ellipse

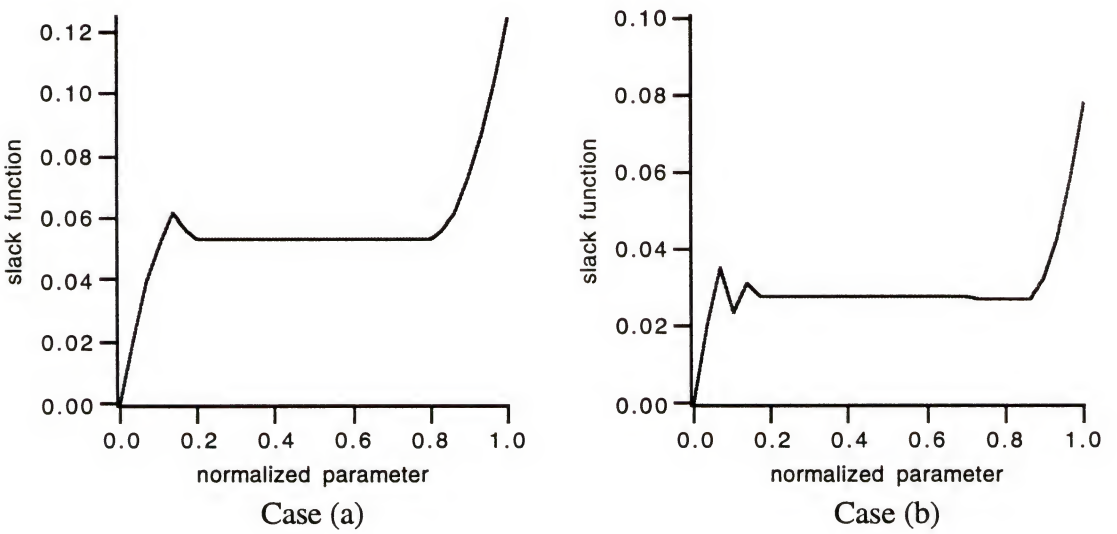


Figure 4.7 Slack Functions for Paths Approaching Symmetry

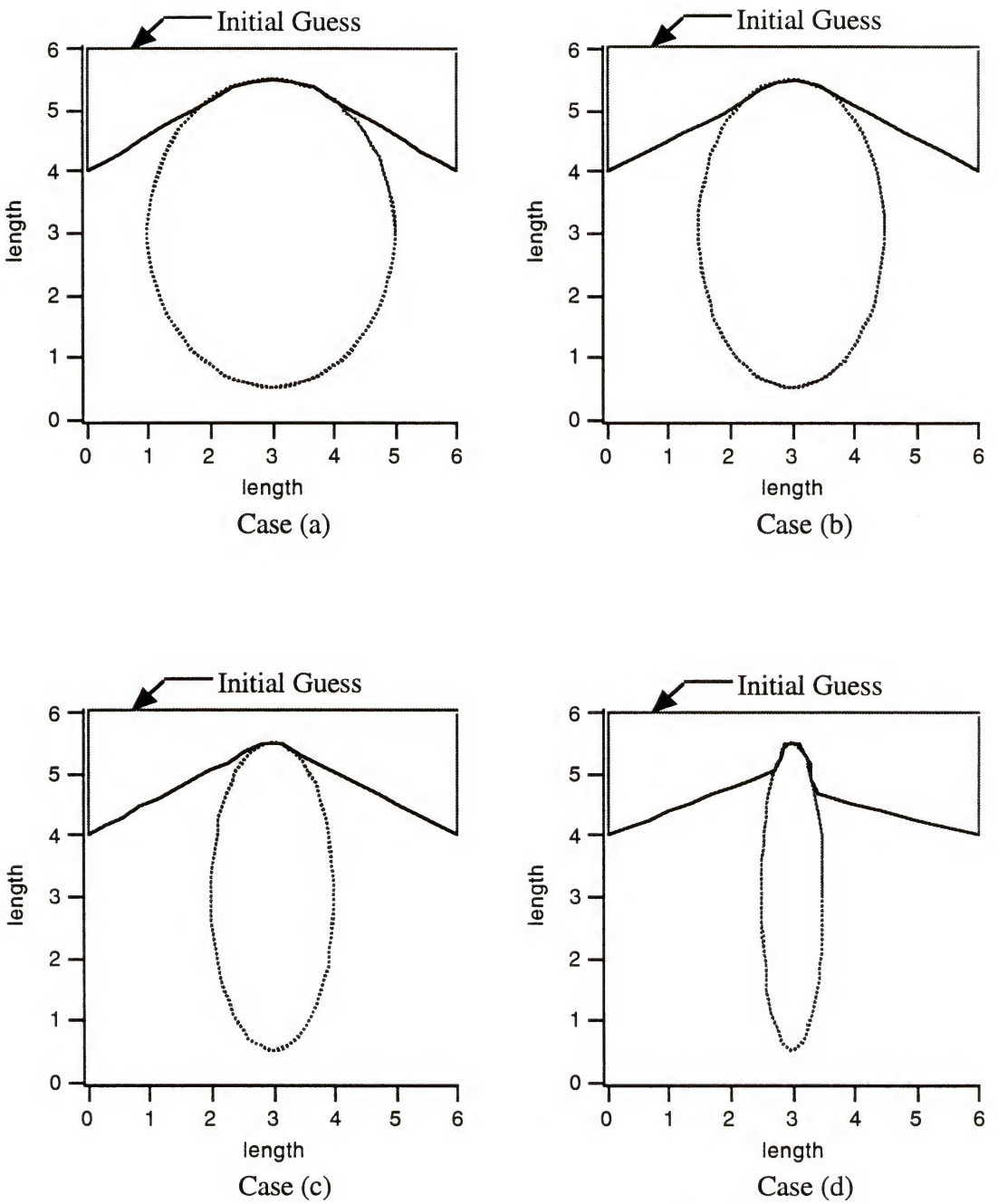


Figure 4.8 Effects of Increasing Curvature on a Solution Path

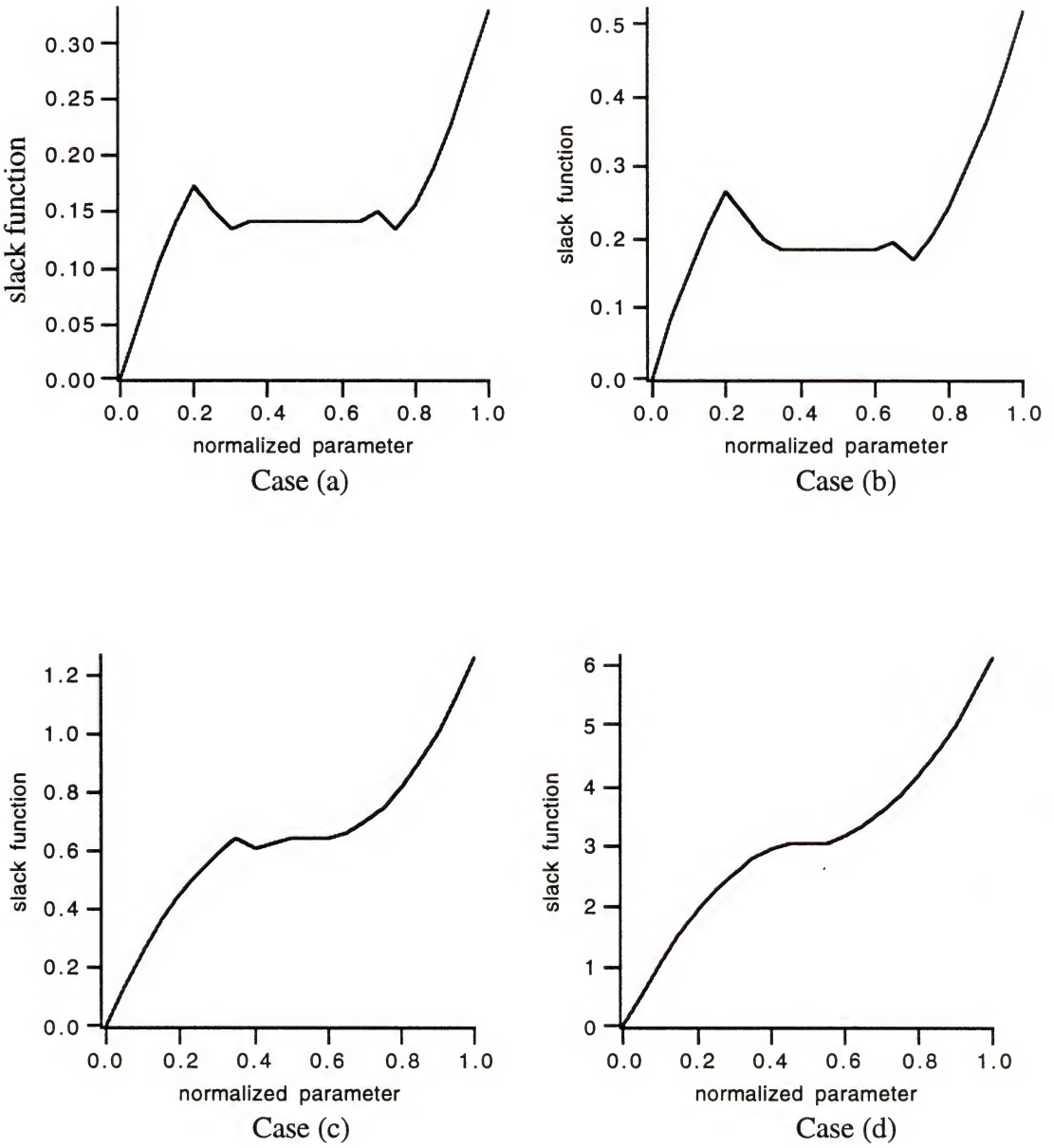


Figure 4.9 Slack Functions for the Example of Increasing Curvature

4.4 Path Planning Around Multiple Obstacles

In this section the path planning algorithm is extended to test its ability to find paths around multiple obstacles. It would be useful to determine before hand if a particular set of obstacles and their geometry lend themselves to the path planning algorithm that has been developed. Unfortunately, this can only be determined by observing the results. In the timing tests performed in Section 4.2 paths were generated for obstacle geometries of up to twelve circles organized in a rectangular array. This can be contrasted to Case (d) in Figure 4.4 where the algorithm fails to find a valid path for bypassing a single ellipse. It is felt that this wide range in performance is caused by the effects the number, shape and positioning of the obstacles have on the construction of the underlying differential equations and on the ability of the algorithm to find solutions to these equations. The examples presented in this section range from an obstacle geometry that is straightforward for the algorithm to converge to a solution, to obstacle geometries for which the algorithm has trouble finding a solution. Throughout these examples the slack functions will be studied to verify that the slack function test provides a good measure of the quality of the solution.

In Figure 4.10 an example is shown for bypassing three circles. As can be seen by examining Table 4.1 the algorithm was tested for numbers of steps from 15 to 20. The solutions were found ranging from 11 to 40 seconds and the paths all have the same length down to the second decimal. The behavior of the algorithm in generating solution paths in this problem can now be contrasted with the solutions generated for the arrangement of six circles shown in Figure 4.11. Over a range of 15 to 25 steps and two different initial guesses, two distinct paths were found labeled as A and B in the figure. As can be seen from Table 4.2 most of the step sizes converge to Path A, but three solutions converge to Path B. Inspection of path B in Figure 4.11 would show it to be a valid, it does not violate any of the obstacle boundaries, with a length only three percent longer on average than the solutions that give Path A. However when the slack functions are examined for the respective paths in Figures 4.12 and 4.13 it can be seen that Path B is an invalid path as judged by using the behavior of the slack functions as a criteria.

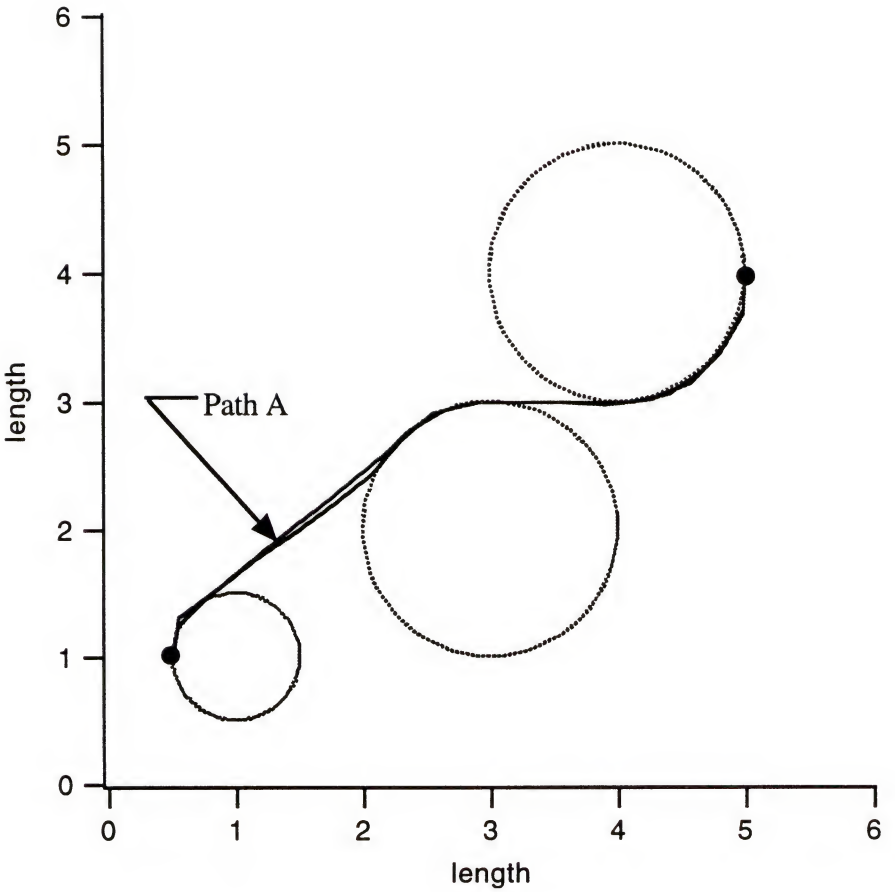


Figure 4.10 Path Planning around Three Circular Obstacles

Table 4.1 Algorithm Performance for Three Obstacle Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	15	120	Path A	29	5.936	11.5
1	16	128	Path A	29	5.935	12.75
1	17	136	Path A	33	5.936	17.45
1	18	144	Path A	33	5.921	21.11
1	19	152	Path A	57	5.916	42.73
1	20	160	Path A	33	5.913	27.11

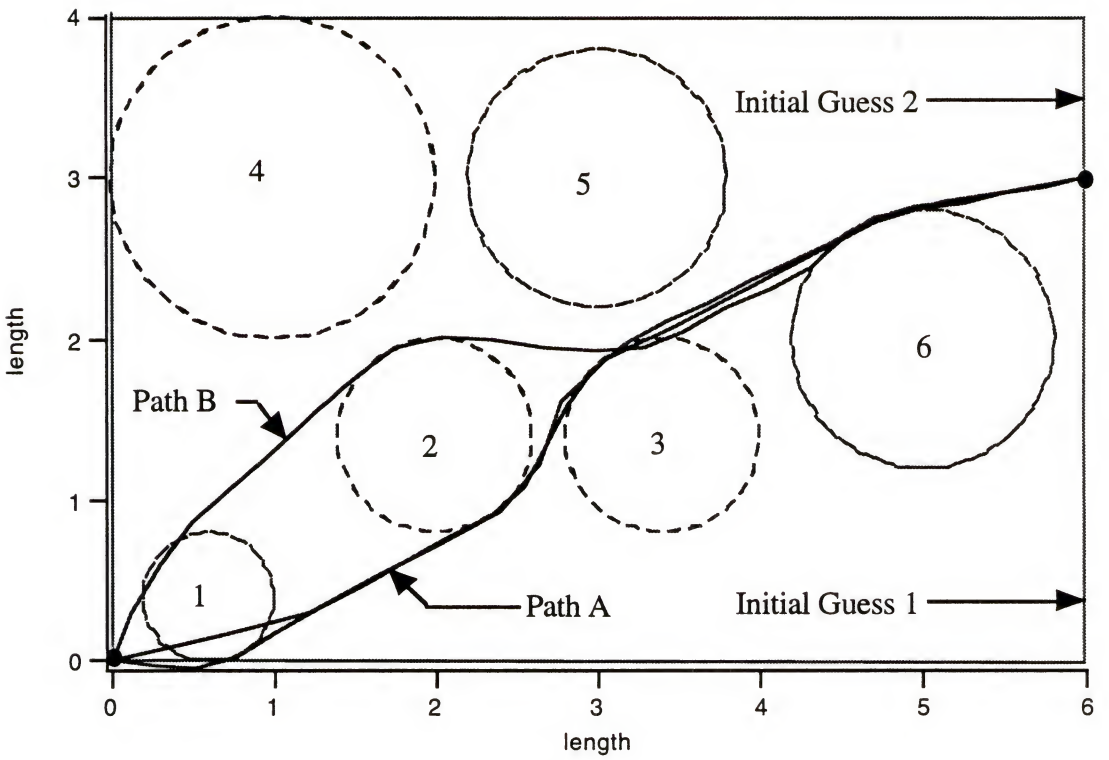


Figure 4.11 Example of Path Planning Around Six Circles

Table 4.2 Algorithm Performance for the Six Obstacle Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU Time (sec)
1	15	210	Path A	26	6.946	48.2
1	16	224	Path A	26	6.987	61.45
1	17	238	Path A	46	6.973	127.75
1	18	252	Path A	75	6.975	264.63
1	19	266	Path A	34	6.979	138.15
1	20	280	Path A	54	6.978	247.16
1	21	294	Path A	29	6.987	157.7
1	22	308	Path A	31	6.982	200.75
1	23	322	Path A	31	6.982	200.41
1	24	336	Path B	50	7.184	420
1	25	350	Path A	30	6.993	298.7
2	15	210	Path A	150	6.983	276.69
2	16	224	Path A	100	7.076	225.04
2	17	238	Path A	35	6.975	96.1
2	18	252	Path A	34	7.007	117.64
2	19	266	Path A	28	6.979	111.92
2	20	280	Path A	32	6.995	144.3
2	21	294	Path B	51	7.177	277.48
2	22	308	Path B	48	7.566	303.77
2	23	322	Path A	35	6.988	252.39
2	24	336	Path A	31	6.988	258.7
2	25	350	Path A	41	7	403.85

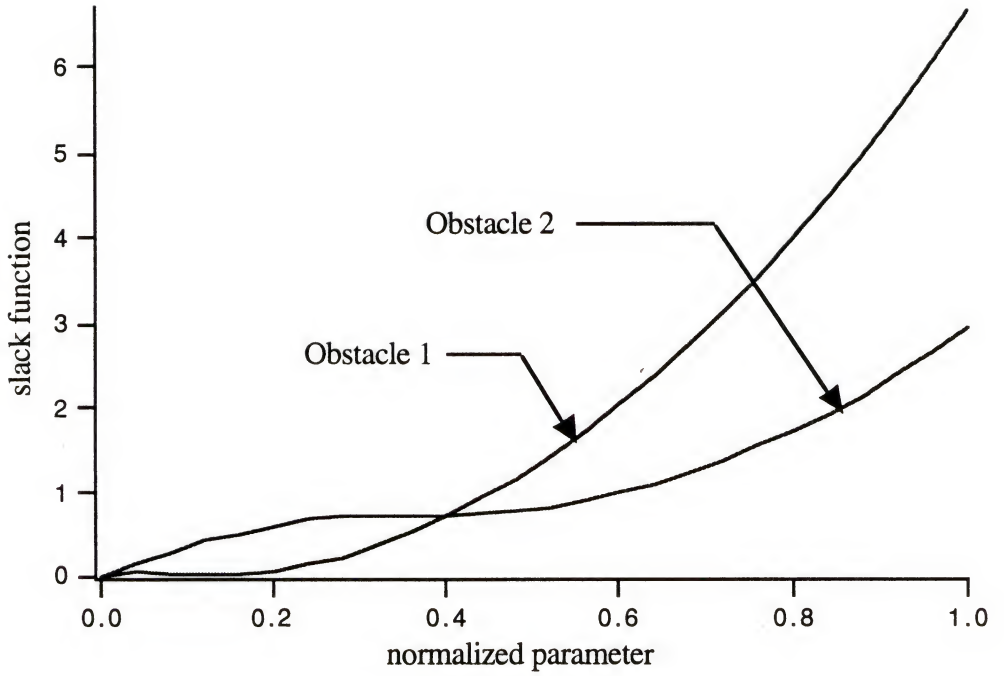


Figure 4.12 Slack Functions for Path A

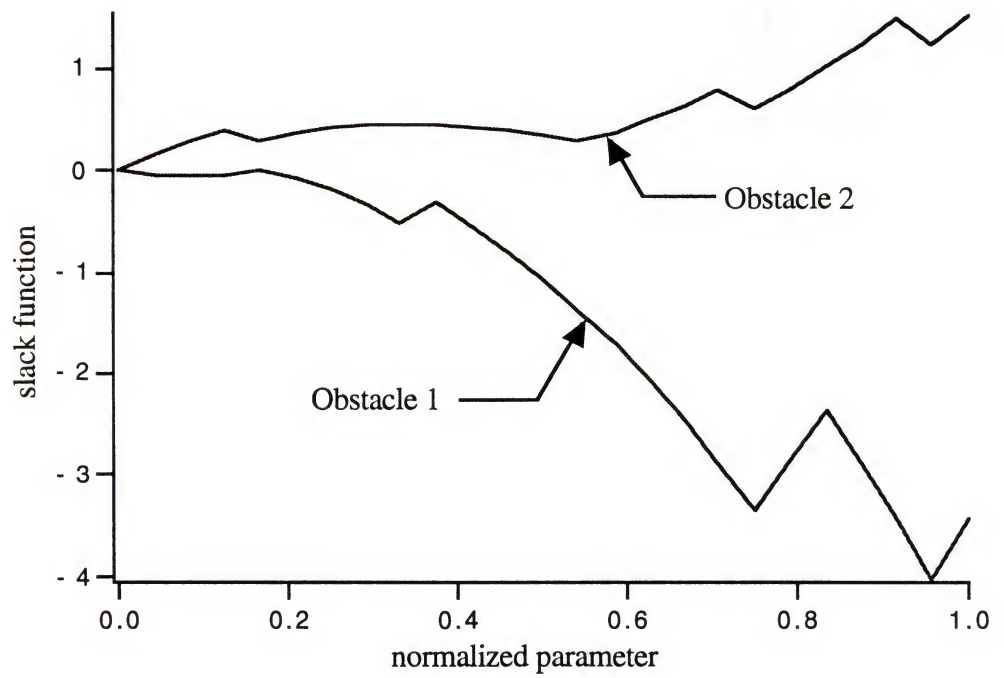


Figure 4.13 Slack Functions for Path B

In Figure 4.14 a passage type problem is shown with two of the solutions, Path A with 15 steps and Path B with 18 steps. The slack functions corresponding to obstacles 2,3, and 5, the obstacles that come in contact with the paths, are displayed in Figures 4.15 and 4.16 respectively. Path B is obviously an invalid path which crosses two obstacles and this is reflected in the smoothness of the solution shown in the slack functions. Various other invalid paths are generated for a path divided into 19 and 20 steps.

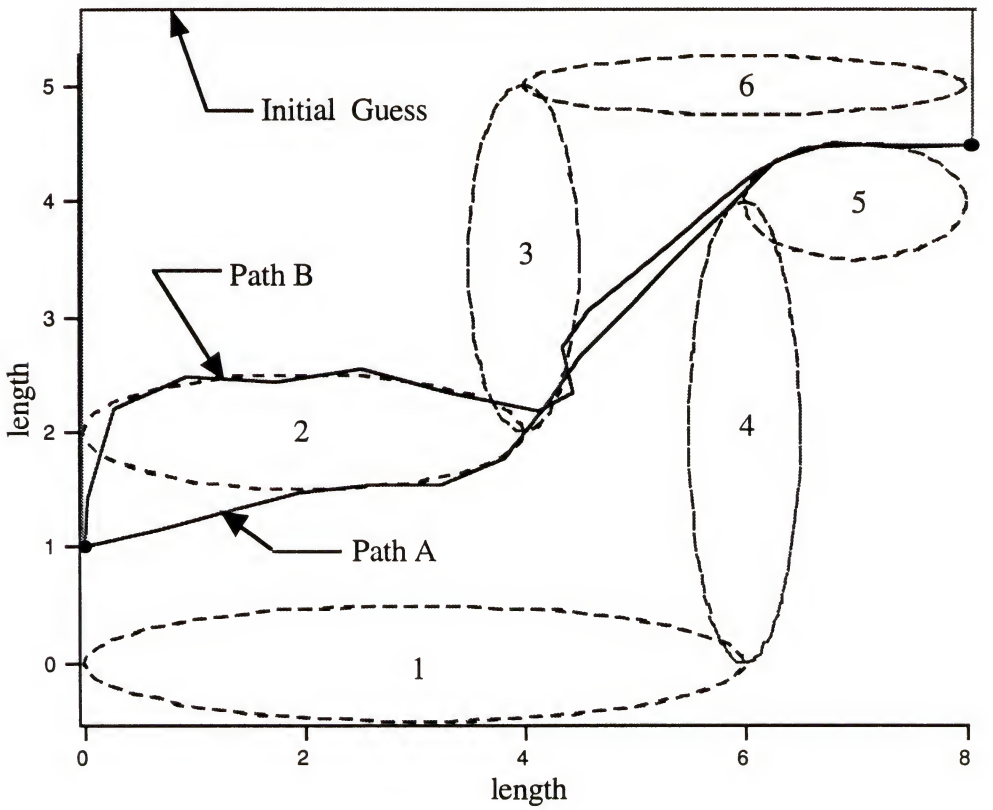


Figure 4.14 Paths Found for Passage Type Problem

Table 4.3 Algorithm Performance for the Passage Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	15	210	Path A	133	9.238	239.8
1	16	224	Path A	85	9.254	195.99
1	17	238	Path A	133	9.26	355.18
1	18	252	Invalid Root	135	10.274	426.91
1	19	266	Invalid Root	161	11.41	609.88
1	20	280	Invalid Root	355	10.822	1590.87

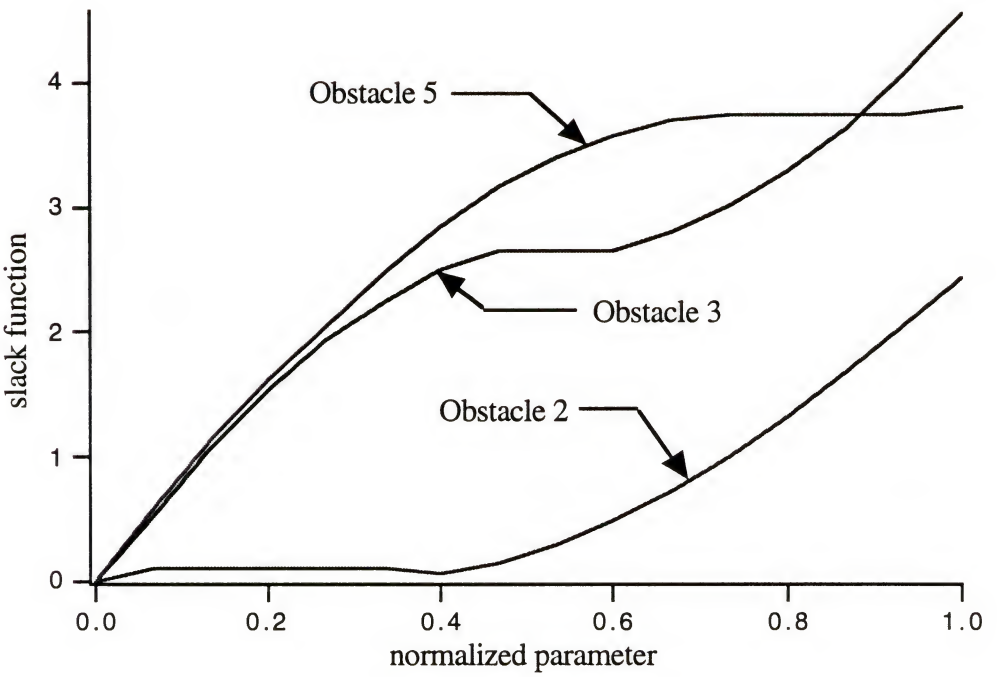


Figure 4.15 Graph of Slack Variables for Solution A

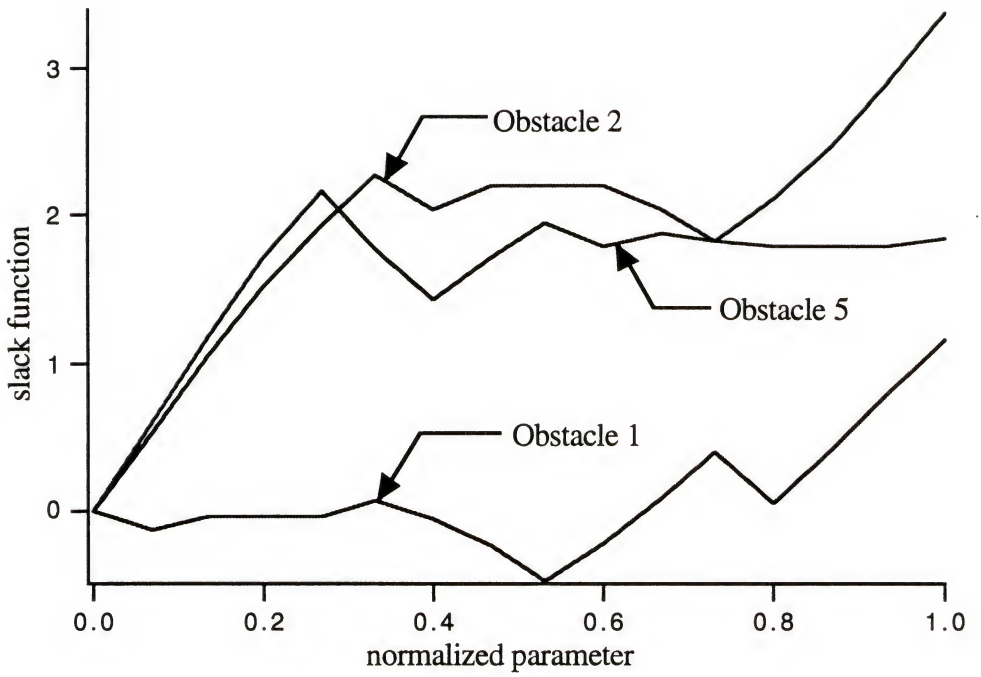


Figure 4.16 Graph of Slack Variables for Path B in the Passage Problem

The purpose of the last two examples is to demonstrate the performance of the algorithm in bypassing a non convex composite obstacle . This will then be compared to paths generated in the next section using stage wise optimization. The first example, shown in Figure 4.17, consists of five circles in a row with symmetry. In this example the algorithm is able to generate a valid path. The slack functions for this example are shown in Figure 4.19. Even though a valid path is generated the slack functions are not smooth and using the test that has been developed this path would be considered invalid. In Figure 4.18 the geometry is complicated even further by deforming the circles into ellipses with sharp curvatures. The path is now invalid, cutting through the corners of obstacles 1 and 3. Once again by examining Figure 4.20 it can be seen that the path would not be considered valid using the slack function test.

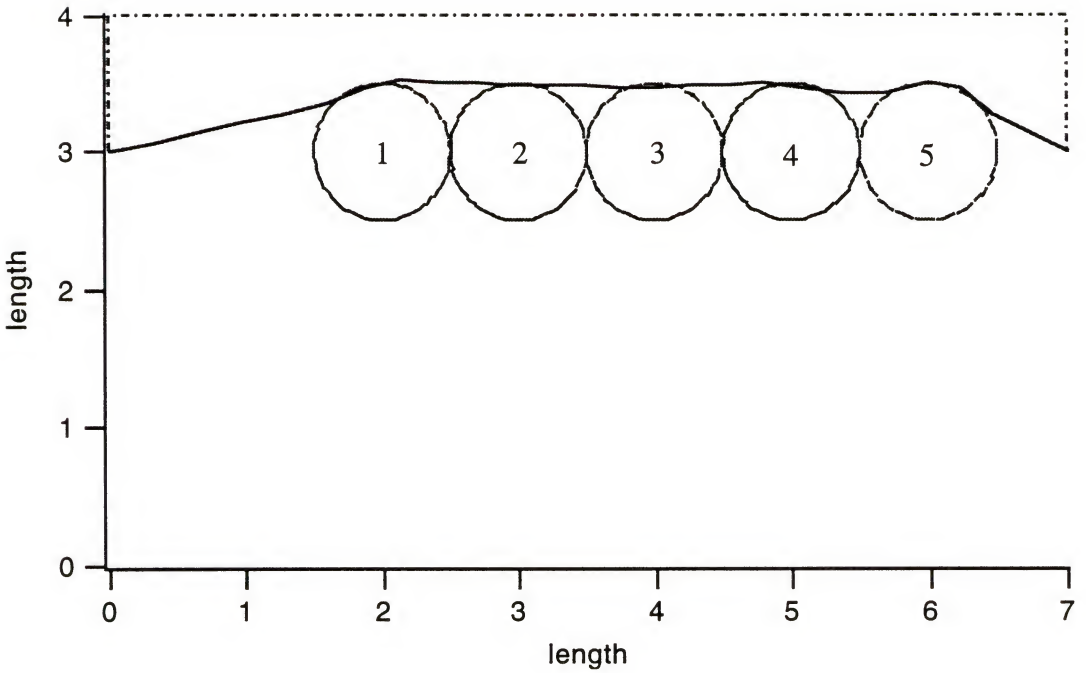


Figure 4.17 Path Planning around a Composite Non convex Obstacle

Table 4.4 Algorithm Performance for Figure 4.17

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	30	360	Path A	132	7.223	786.29

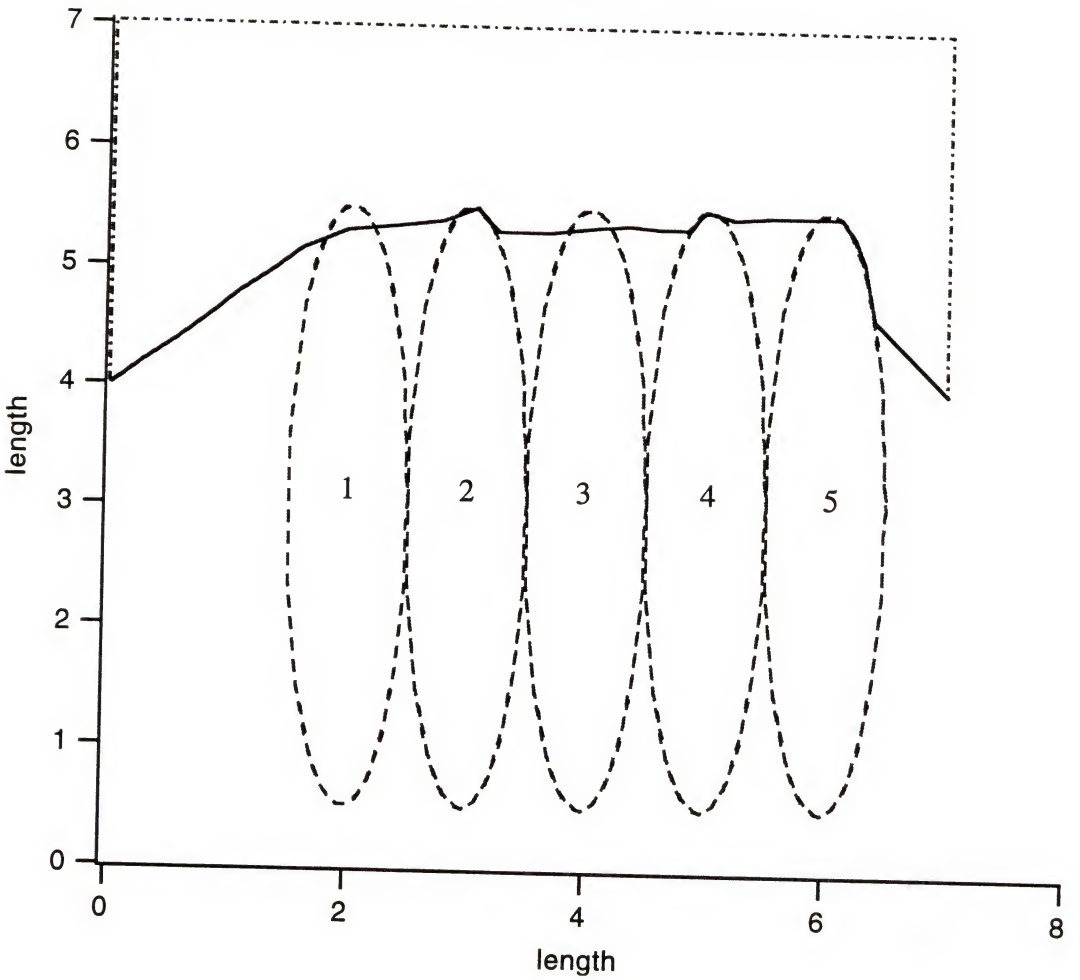


Figure 4.18 A Path around a Non convex Obstacle

Table 4.5 Algorithm Performance for Figure 4.18

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	30	360	Path A	167	8.476	1845.69

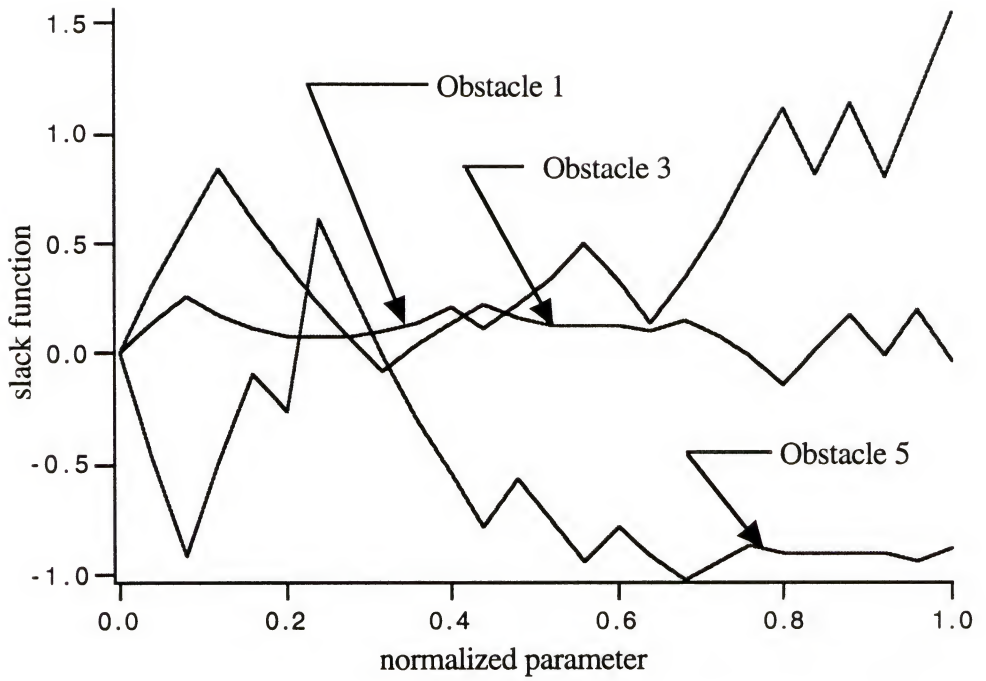


Figure 4.19 Slack functions for circular obstacles

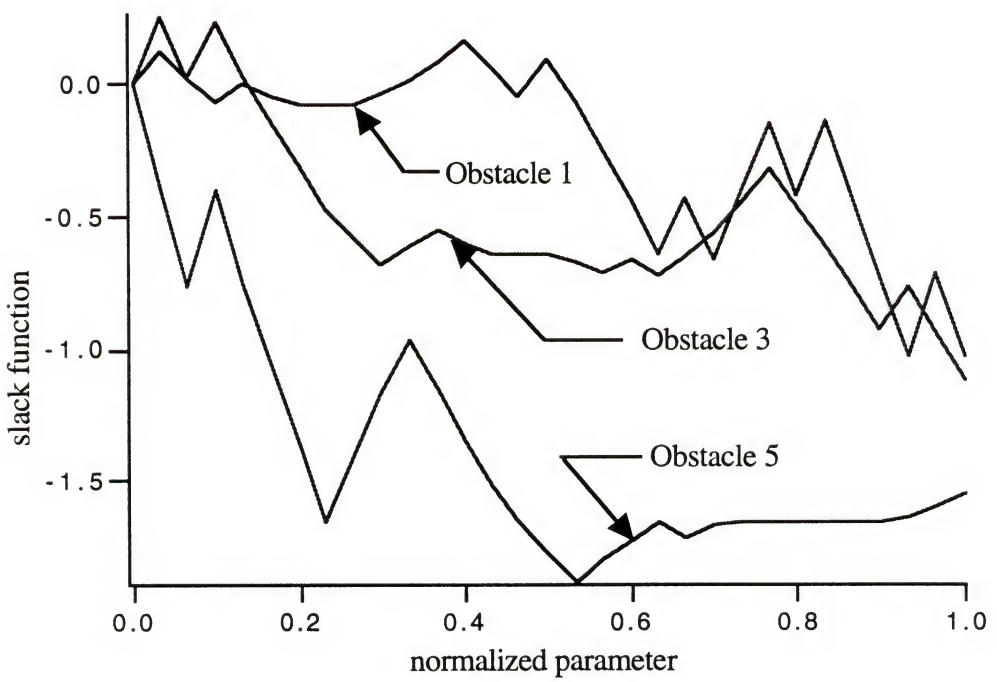


Figure 4.20 Slack functions for elliptical obstacles

4.5 Path Planning Using Stage Wise Optimization

In the previous section several examples were shown that demonstrate the performance of the algorithm under a variety of obstacle geometries. As these tests became more complicated it was observed that some of the solutions generated by the algorithm produced invalid paths. It was further observed that the paths could be quickly judged by looking at the smoothness and form of the slack functions in the solution. This test gives a good indication of the quality of the solution for most cases. There are some cases, such as Path B Figure 4.11 and the path generated in Figure 4.17, where a valid solution which does not violate any obstacle boundaries and is efficient would be rejected. However, in all the cases that have been examined the test does not judge an invalid path to be valid. The next step is to examine possible methods for correcting these invalid paths. The first possible method is that when a solution is generated which does not pass the slack function test, increase or decrease the number of steps in the problem and rerun the algorithm. For example, in Figure 4.11, the algorithm is initially run for 24 steps and initial guess 1 where Path B was generated. If the number of steps is increased by one step the algorithm will converge to a valid solution. The overall CPU time to find a valid path would be 718.7 seconds. A similar result could be obtained by decreasing the step size.

A second possible method is the stage wise optimization algorithm which was introduced in Chapter 3. The improvement in the performance of the algorithm can be attributed to finding a good solution with only one obstacle present. The solution is then used as the initial guess in the next step of the algorithm when the next obstacle is added and so forth. When the overall problem is considered in the final step, a refined initial guess based on information obtained in the preceding steps is used. The weakness of the algorithm is that it does not deal directly with the causes of instability in the original problem. It only tries to avoid instabilities in each step by having an initial guess close to the solution. There is no guarantee that an instability will not occur. However by going through the process of only adding one obstacle at a time, the probability of finding an invalid root is greatly reduced.

The performance of the stage wise optimization will now be examined by rerunning the cases that failed in the previous section. As will be seen the paths that are generated are valid but the paths are not always smooth. The greatest improvement is in the slack functions which now behave in the ideal manner described in Chapter 3.

A stage wise optimization for the six circle problem is shown in Figure 4.21. This example has been run for 24 steps and initial guess 1, corresponding to the first solution that generates path B in Table 4.2. As can be seen the path is not as smooth as those in Figure 4.11 but a valid path has been generated. The slack functions for this example are shown in Figure 4.22 and can be directly compared with those in Figure 4.13.

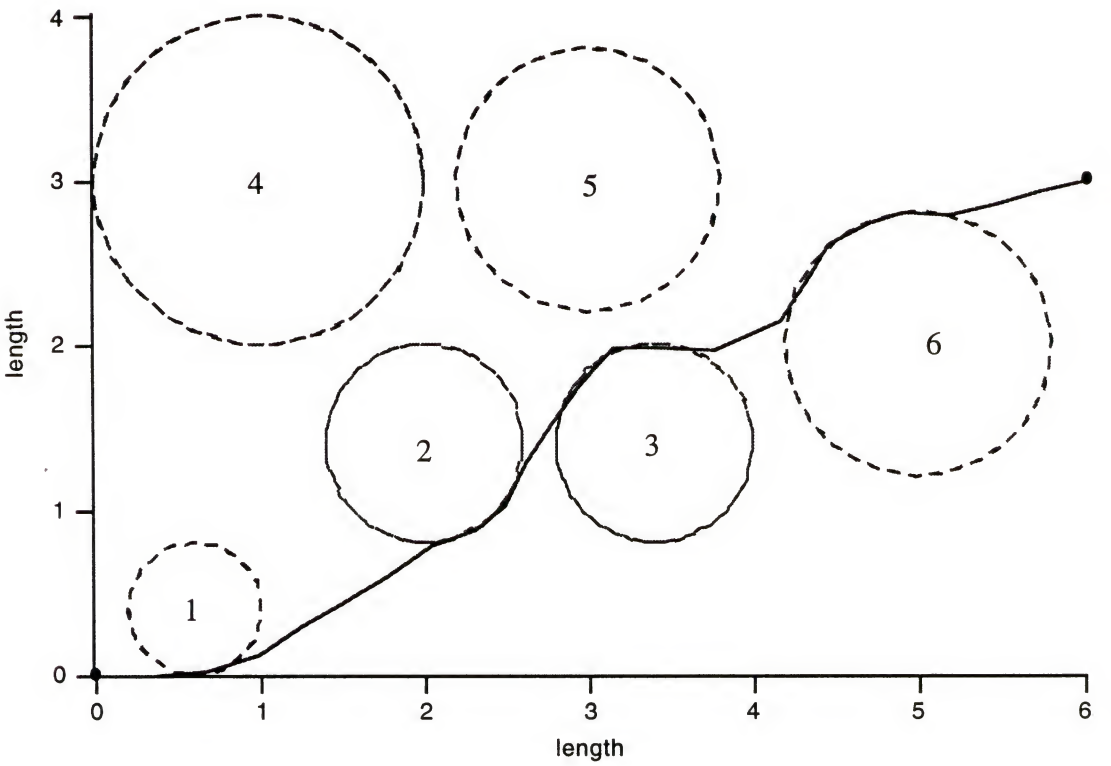


Figure 4.21 Solution Path for Stage Wise Optimization of Six Obstacles (24 Steps)

Table 4.6 Algorithm Performance For Stage Wise Optimization of the Passage Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	18	252	Path A	90	7.17	257.27

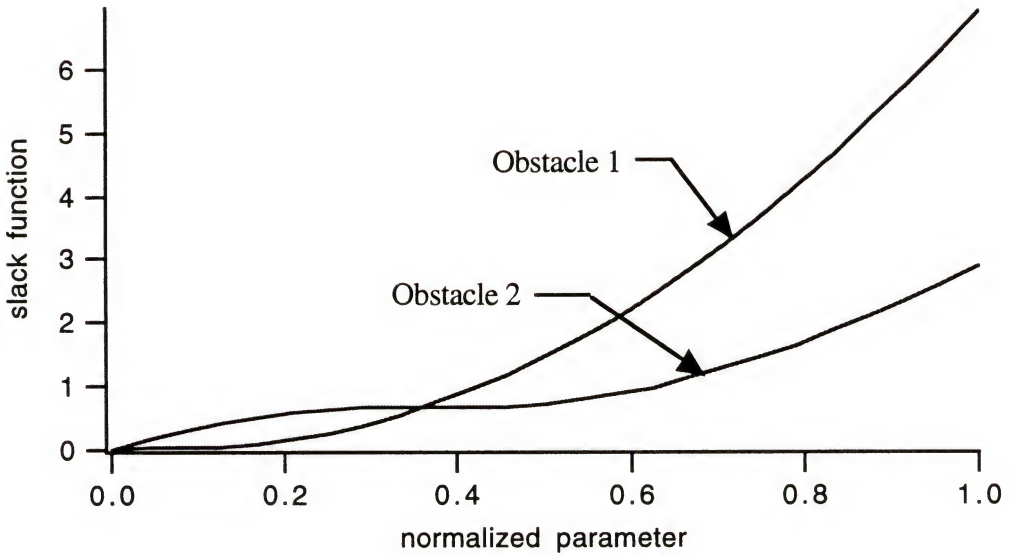


Figure 4.22 Slack Functions for Stage Wise Optimization of Six Obstacles

In Figure 4.23 the passage problem is shown with a stage wise optimization for 18 steps. Table 4.7 can be directly compared with Table 4.3. The stage wise optimization algorithm takes approximately twice as long to find a solution than the corresponding problem using regular optimization. The slack functions in Figure 4.24 can be directly compared to the slack functions in Figure 4.16.

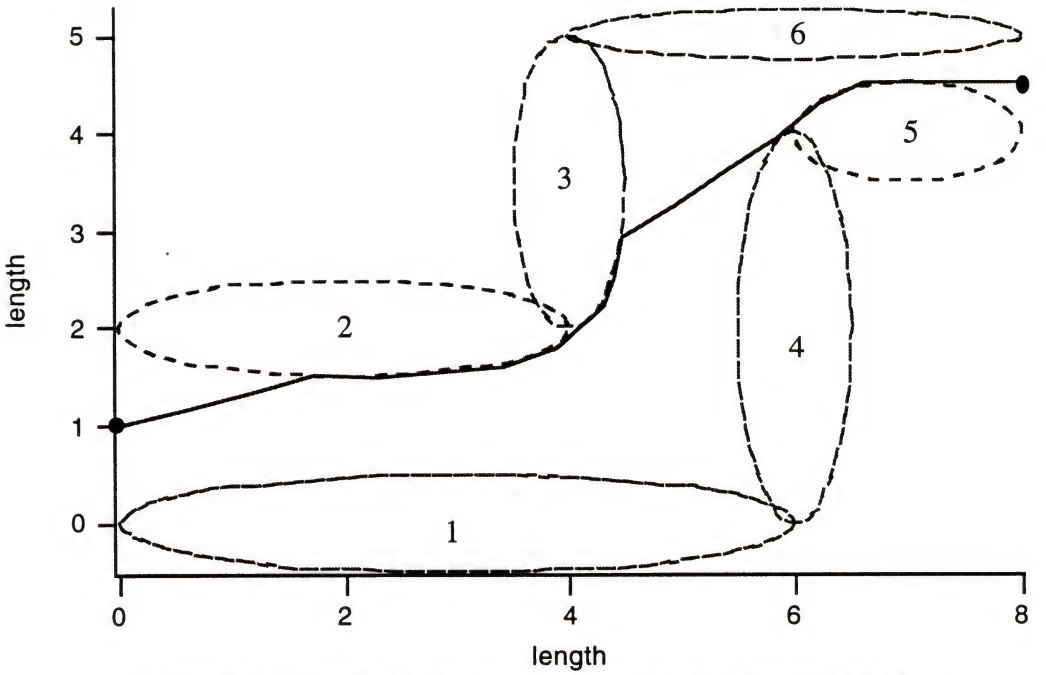


Figure 4.23 Stage Wise Optimization for the Passage Problem

Table 4.7 Algorithm Performance For Stage Wise Optimization of the Passage Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	18	252	Solution A	754	9.371	924.49

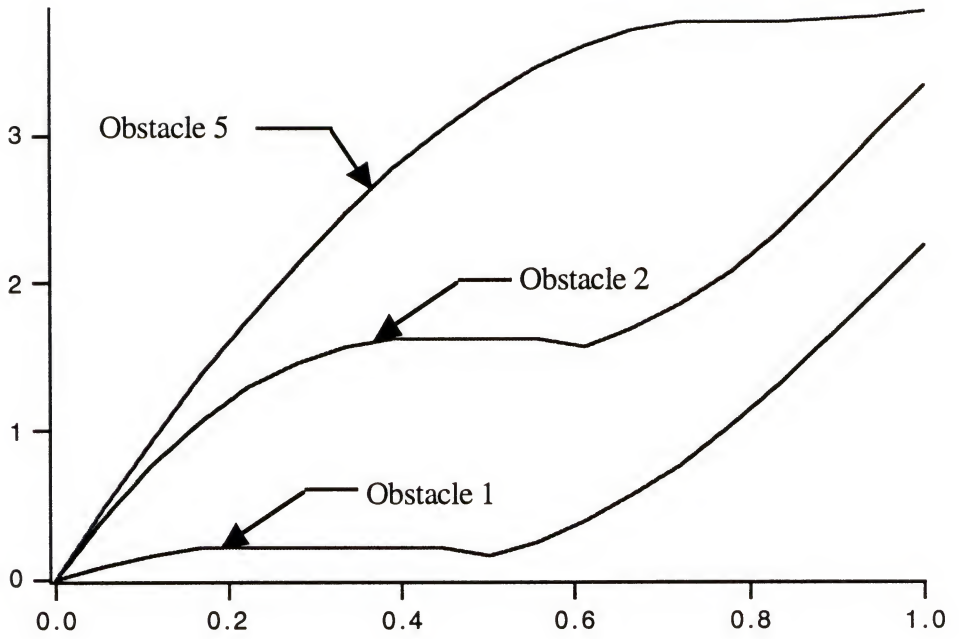


Figure 4.24 Slack Functions for the Stage Wise Optimization of the Passage Problem

The previous two examples have shown improvement in both the path and the slack functions by using a stage wise optimization. In the next two examples involving the composite non convex obstacles, this is not the case. In Figure 4.25 the path generated is not as smooth as the path in Figure 4.17, but the stage wise algorithm has generated smooth slack functions as can be seen in Figure 4.27. In Figure 4.26 there is a slight change in the modified path over the path in Figure 4.18, but once again the stage wise algorithm has generated smooth slack functions. This can be seen by comparing Figure 4.28 with Figure 4.20. This would seem to suggest that in some instances the stage wise algorithm has more success in generating smooth slack functions than in generating a smooth path.

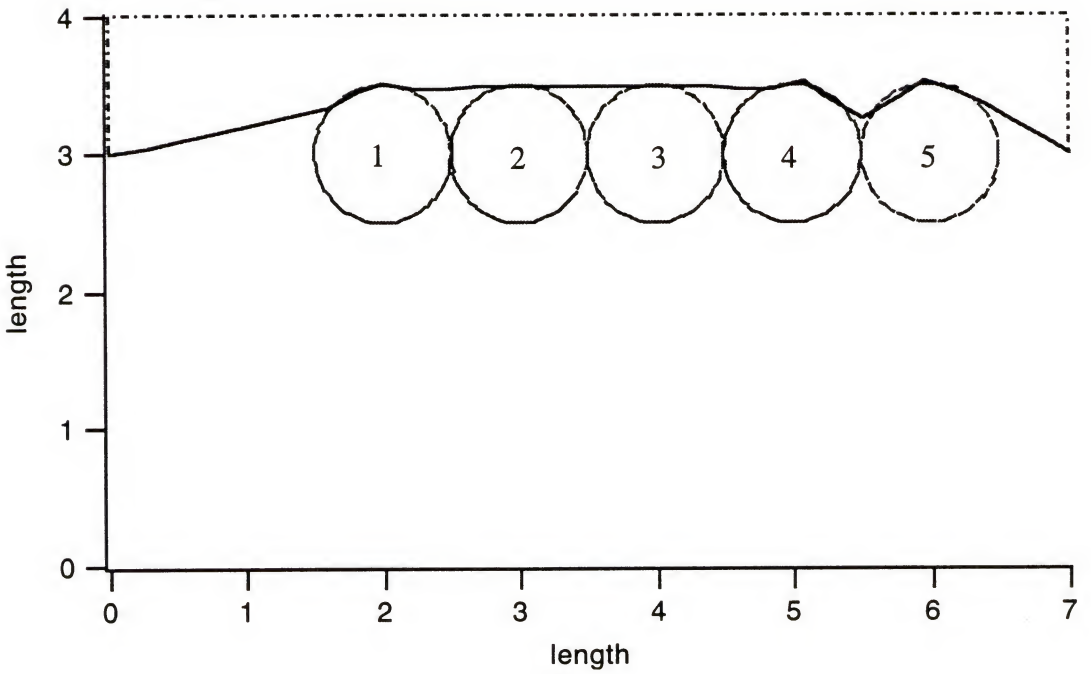


Figure 4.25 Stage Wise Optimization of the Circle Problem

Table 4.8 Algorithm Performance For Stage Wise Optimization of the Circle Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	30	360	Solution A	104	7.381	292.42

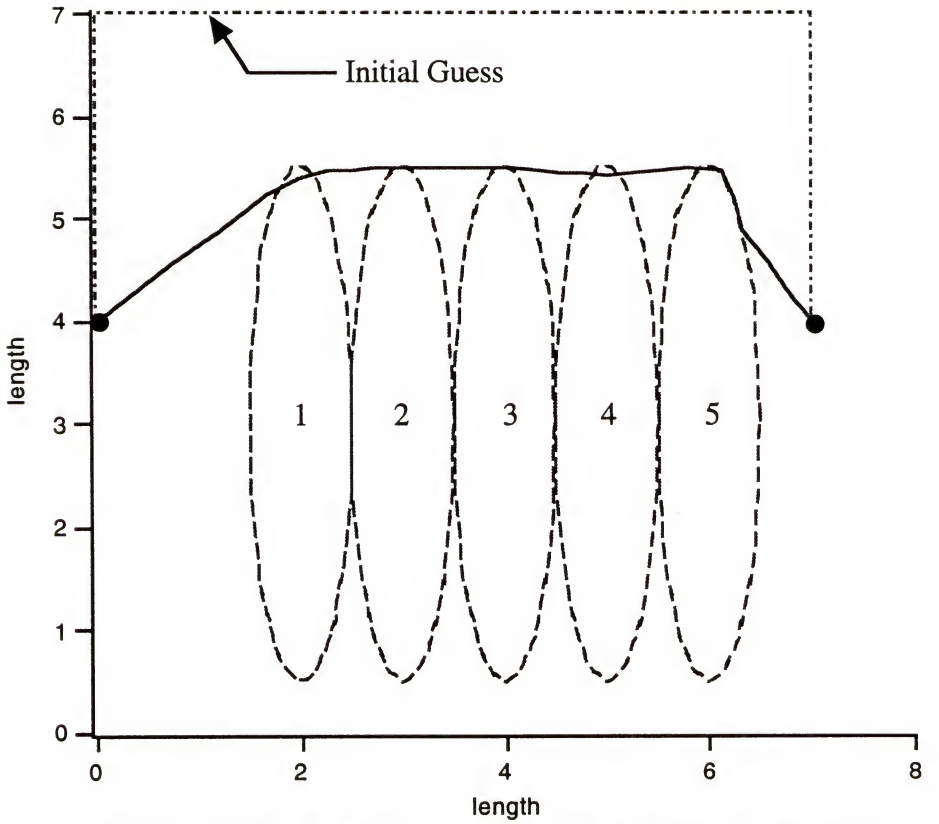


Figure 4.26 Stage Wise Optimization of the Elliptical Problem

Table 4.9 Algorithm Performance For Stage Wise Optimization of the Elliptical Problem

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	30	360	Solution A	278	8.318	386.28

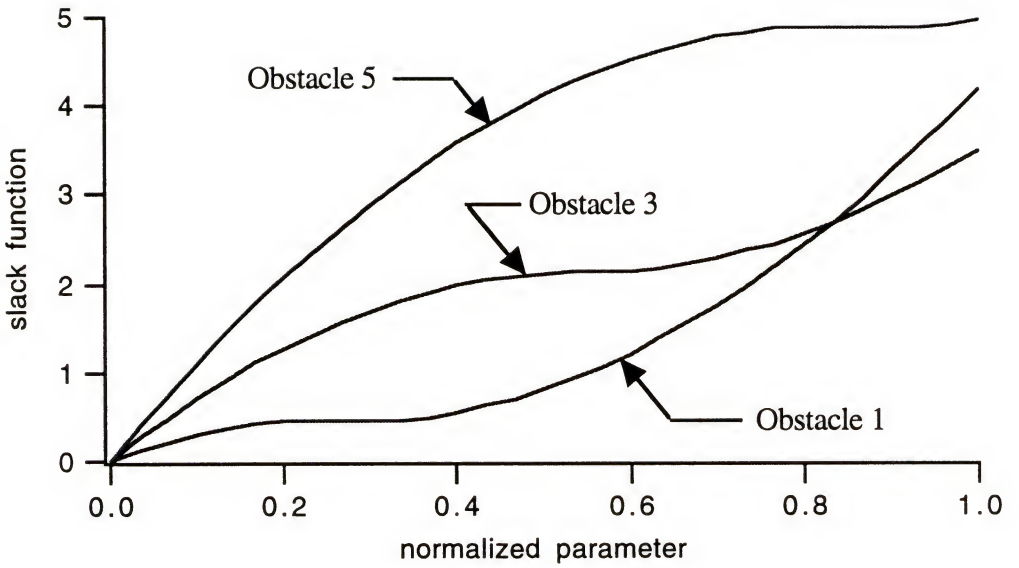


Figure 4.27 Slack Functions for the Stage Wise Optimization of the Circle Problem

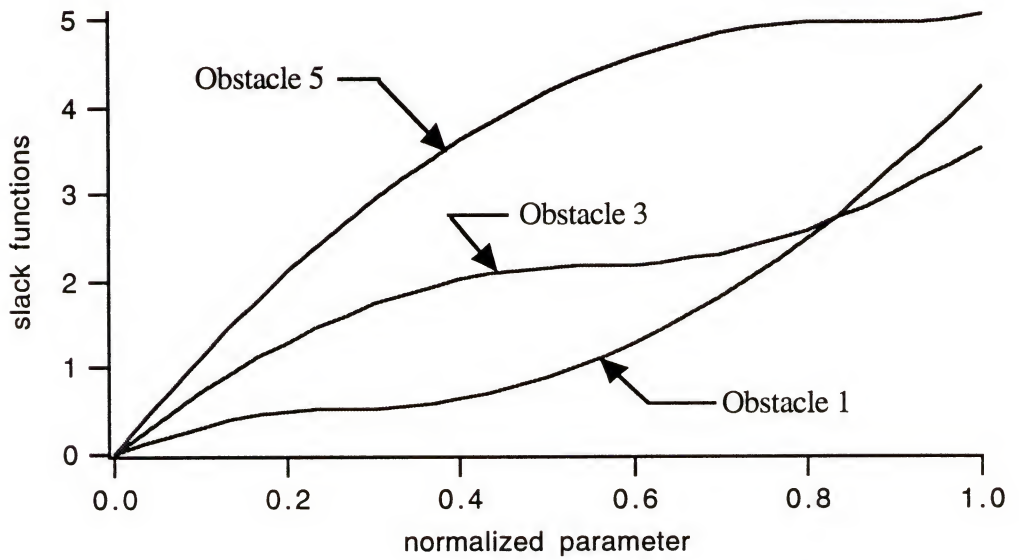


Figure 4.28 Slack Functions for the Stage Wise Optimization of The Elliptical Problem

4.6 Path Planning Around Moving Obstacles

The final example in this chapter will demonstrate path planning around an obstacle moving from left to right on a horizontal path. Unfortunately the solutions generated by the algorithm for this type of problem are unstable. For paths with step sizes ranging from 14 to 20 none of the resultant paths are the same. All vary in some way. Figures 4.29 and 4.30 show the two best paths that are found out of all the test runs tried in Table 4.10. Their corresponding slack functions are shown in Figures 4.20 and 4.21 respectively. Because there is only one obstacle in this example it is not possible to use the stage wise optimization algorithm to improve performance.

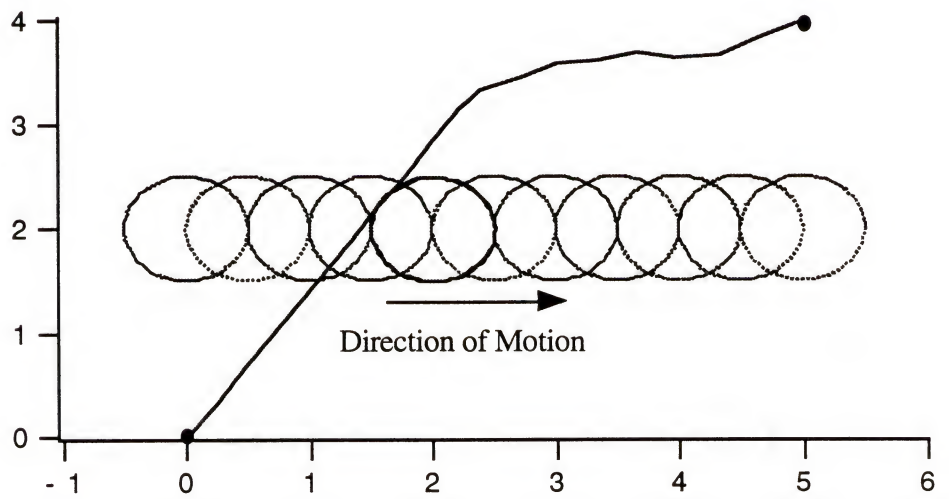


Figure 4.29 Path Generated for a Solution with 19 Steps

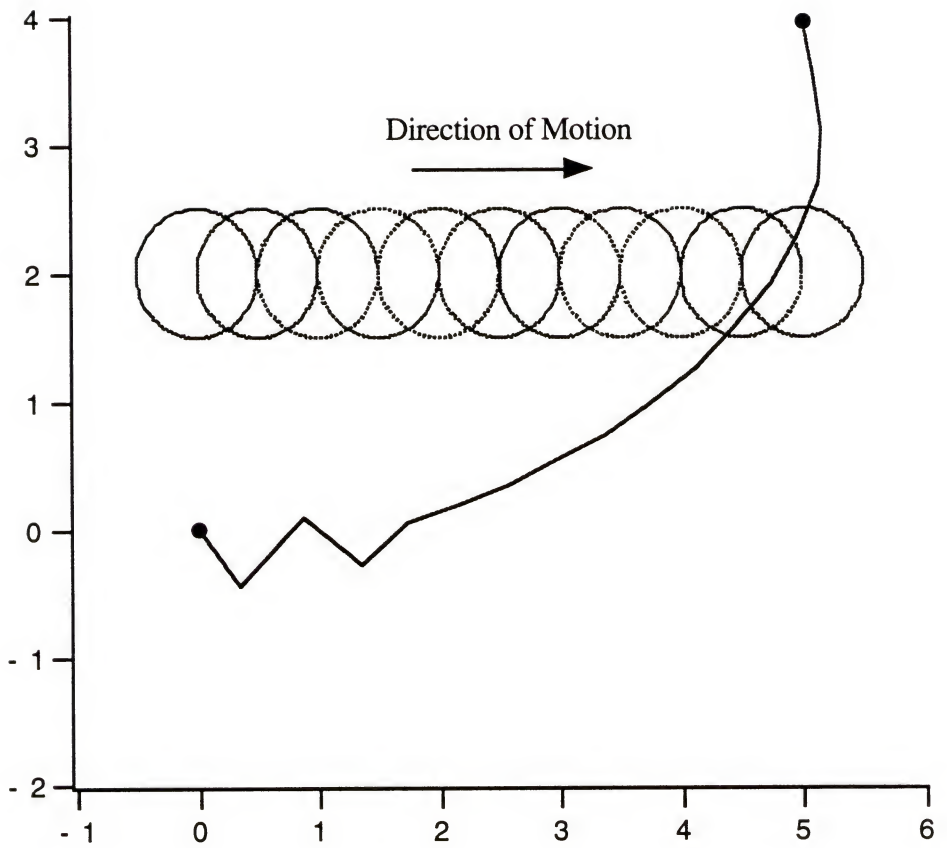


Figure 4.30 Path Generated for a Solution with 17 Steps

Table 4.10 Statistics for Moving Obstacle Example.

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	14	56		102	7.799	5.02
1	15	60		36	6.69	1.99
1	16	64		116	13.702	7.56
1	17	68		151	8.198	11.59
1	18	72		150	13.79	15.17
1	19	76		124	6.853	12.65
1	20	80		135	8.054	15.92

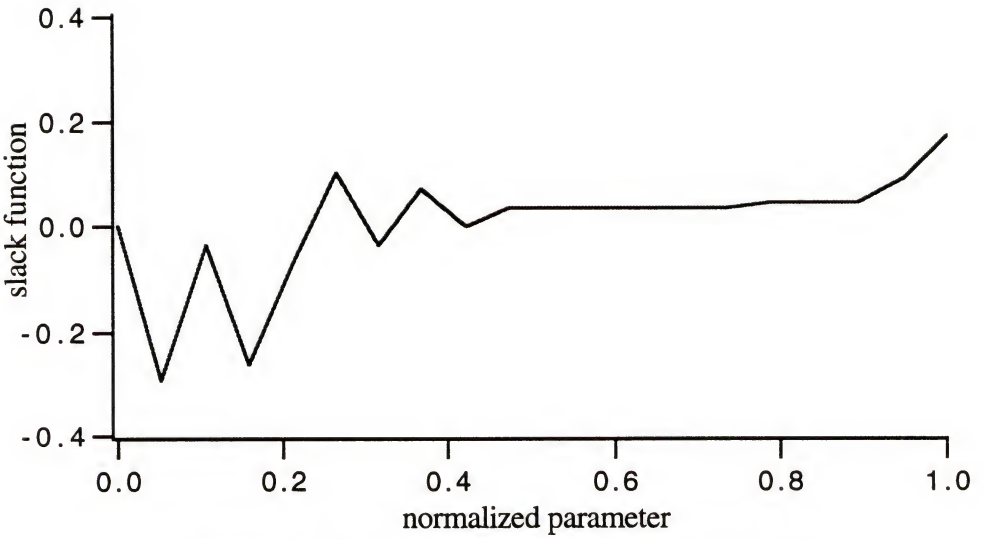


Figure 4.31 Slack function for Path in Figure 4.30

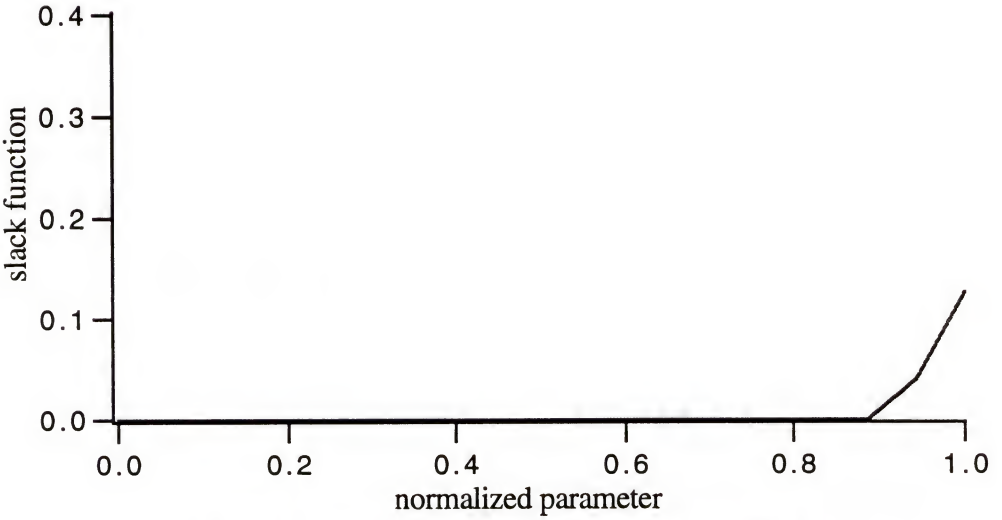


Figure 4.32 Slack Function for Path in Figure 4.31

CHAPTER 5
PATH PLANNING IN EUCLIDEAN SPACE

5.1 Selection of Test Cases

This chapter demonstrates the extendibility of the algorithm into three-dimensional, Euclidean space. It will be shown that the path planning algorithm that has been developed can be applied to spatial path planning without any modification. This is in sharp contrast to some of the other path planning methodologies that were surveyed in Chapter 1. In general, path planning in three dimensions is much more complicated than the corresponding planar cases covered in the previous chapter. Firstly, it is much more difficult to construct obstacle geometries that will test the algorithm due to the fact that the path is a one-dimensional curve moving around a three-dimensional obstacle. As obstacles are positioned to deflect the path, a new solution is generated which slips around the surface of the obstacle. For instance in Figure 4.8 a series of tests were performed checking the effects of obstacle curvature on the generated path and because this was done in the plane the algorithm was forced to find a path around the increasing curvature. In the three-dimensional case there is no analogue. In Figure 5.4 the path simply curves around the surface of the ellipsoid missing the sharp curvature at the end of the obstacle. This type of behavior makes it virtually impossible to systematically test the algorithm. Thus the examples in this chapter will be more illustrative in nature showing the abilities of the algorithm. As a measure of the efficiency of the generated path, its length will be compared against the shortest distance between the start and end points in each problem. Secondly, the effects of path planning in three-dimensions can be seen on the quality of the solutions that are generated. The slack function test that was developed in Chapters 3 and 4 no longer works for spatial path planning. As will be seen smooth solution paths are

generated without correspondingly smooth slack functions. It is felt that this is directly due to the increase in dimension adding to the complexity of the underlying system of differential equations.

In Section 5.2 timing tests are performed in a similar manner to those conducted for the planar case. The only difference is that an upgraded CPU is used which runs at 150 MHz. Several examples are presented demonstrating the ability of the algorithm to find efficient paths around configurations of spheres and ellipsoids. The tests are presented in Section 5.3. In Section 5.4 two examples are shown testing the performance of the stage wise algorithm.

5.2 Timing Tests

In general the CPU time increases with the complexity of the problem as was observed in the planar case. The complexity of the problem consists of many factors including the number of obstacles, their shape and their relative position thus making it hard to quantify. In the tests that were performed, it was attempted to standardize these factors as much as possible. The number of obstacles is directly related to the matrix size n in the problem and is given by the formula n = (N + 2*M)*P, where N is the number of dimensions, M is the number of obstacles, and P is the number of steps in the path. The matrix dimension gives the best measure of the complexity of the problem in terms of the number of obstacles. For the largest problem tested, the corresponding matrix dimension was 780. To further standardize the tests, the geometry of the obstacles was fixed into a regular array to standardize the positioning and shape of the obstacles. Tests were run on arrays ranging from four through eighteen spheres. They were run on a Silicon Graphics VGX with a MIPS R4000 CPU running at 150 MHz. Figures 5.1a and 5.1b show two different views of the largest example considered, consisting of eighteen spherical obstacles. This matrix is inverted in each iteration of the Newton-Raphson algorithm. The number of iterations varies widely for the problems and no general trend can be discerned in Figure 5.2. However, in Figure 5.3 there is a general increase in CPU time with a corresponding increase in matrix size.

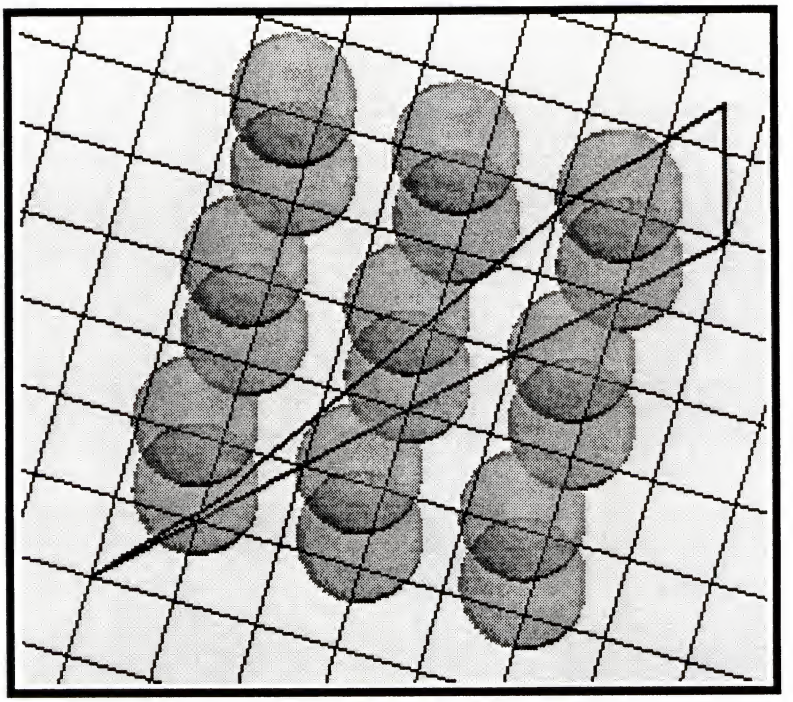


Figure 5.1a Path Planning Around 18 Obstacles

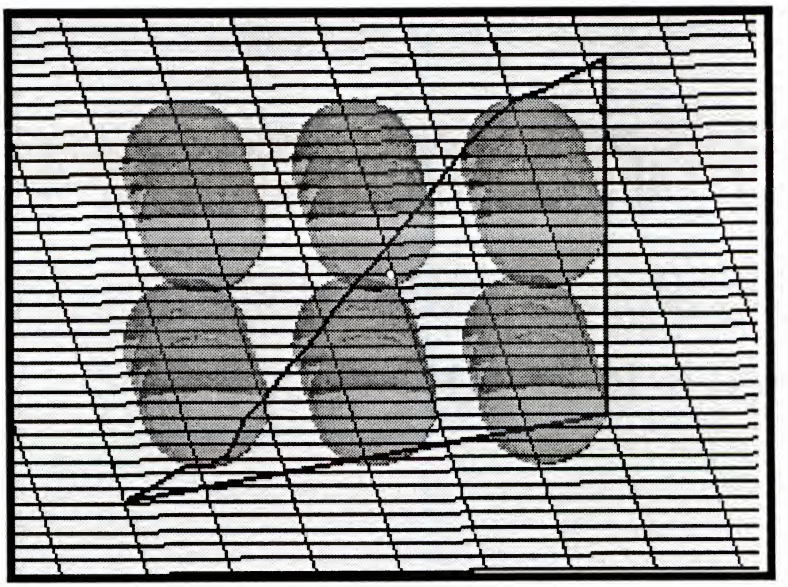


Figure 5.1b Path Planning Around 18 Obstacles

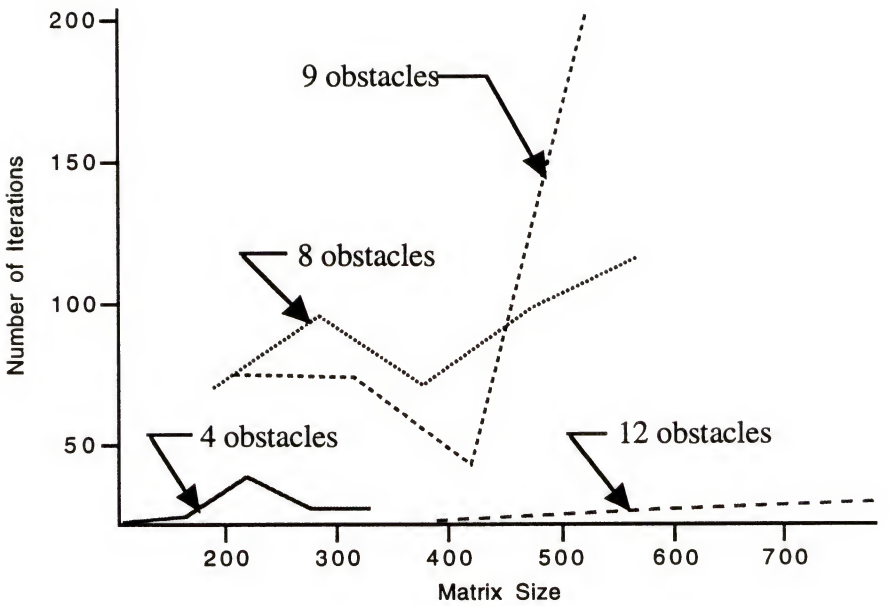


Figure 5.2 Number of Iterations vs Matrix Size for Three-Dimensional Cases

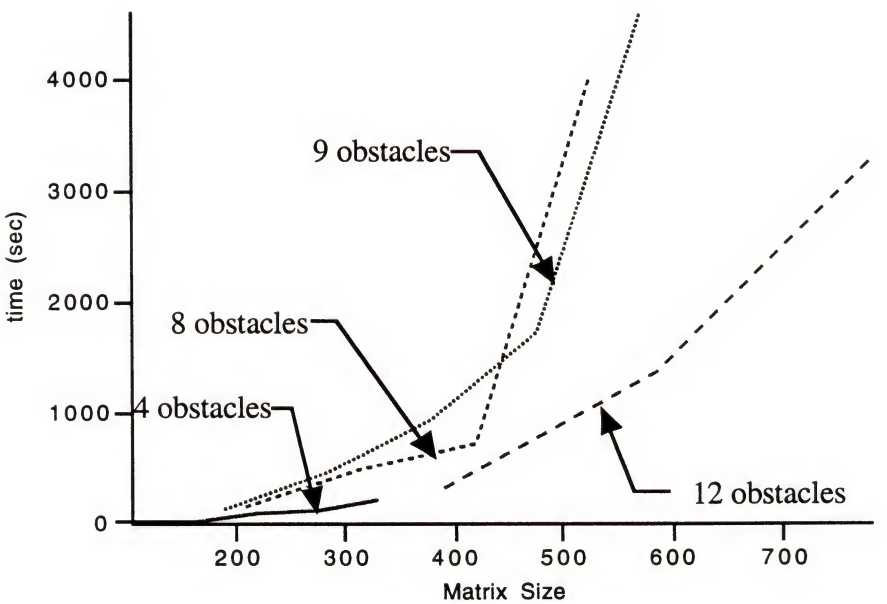


Figure 5.3 CPU Time vs Matrix Size for Three-Dimensional Cases

5.3 Path Planning around Spherical and Ellipsoidal Obstacles

The following three examples demonstrate the ability of the algorithm to generate paths around general configurations of obstacles in Euclidean space. The solution paths that are generated are smooth and efficient. However, in the examples, the slack functions are not smooth and do not always behave in the idealized manner described in Chapter 3. This is a reflection of the increase in complexity of the path planning problem in three dimensional space.

In the first example shown in Figure 5.4 the algorithm generates a path around a single ellipsoid. The starting point is at coordinates (0,0,0) and the end point at coordinates (4,4,6) giving a distance between the two points of 8.246. In comparison the length of the path found around the obstacle 9.343. The slack function for the obstacle is given in Figure 5.5 and does not have the idealized behavior.

In Figure 5.6 the algorithm is tested for two intersecting ellipsoids. The starting point is at coordinates (0,0,0) and the end point at coordinates (4,3,4) giving the distance between the two points as 6.403. The algorithm finds a path around the ellipsoids of length 6.861. The corresponding slack functions are shown in Figure 5.7 and while there are some sharp edges, the quality of the slack functions is not a smooth as those generated smooth paths in the plane.

In the final example, shown in Figure 5.8, in this section the algorithm plans a path around a seven spheres of varying sizes. The starting point is at (0,0,0) and the end point is at (8,7,6) giving a distance for the two points of 12.2. In comparison the length of the path generated by the algorithm is of length 13.319. The slack functions, presented in Figure 5.9 are shown for obstacles 1,2, and 6, which are the obstacles that come in contact with the path.

In general it can be stated that the algorithm is successful in generating paths around arbitrary configurations of obstacles. As will be seen by an example in the next section the algorithm fails in cases when the curvature of the obstacles becomes too sharp. It is also evident that the slack function test is no longer reliable in even the simplest cases where smooth efficient paths are generated.

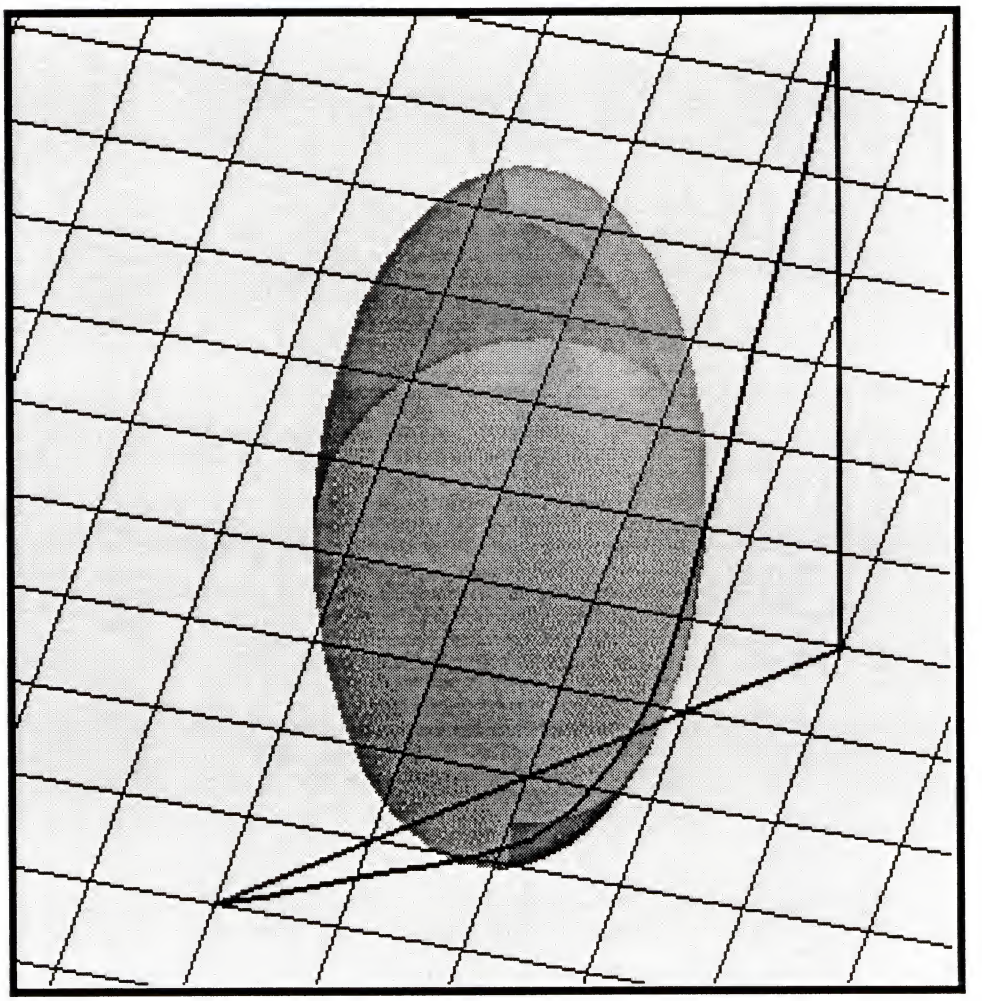


Figure 5.4 Path Planning around an Ellipsoidal Obstacle

Table 5.1 Algorithm Performance for Path Planning around an Ellipsoidal Obstacle

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	20	360	Solution A	66	9.343	14.4

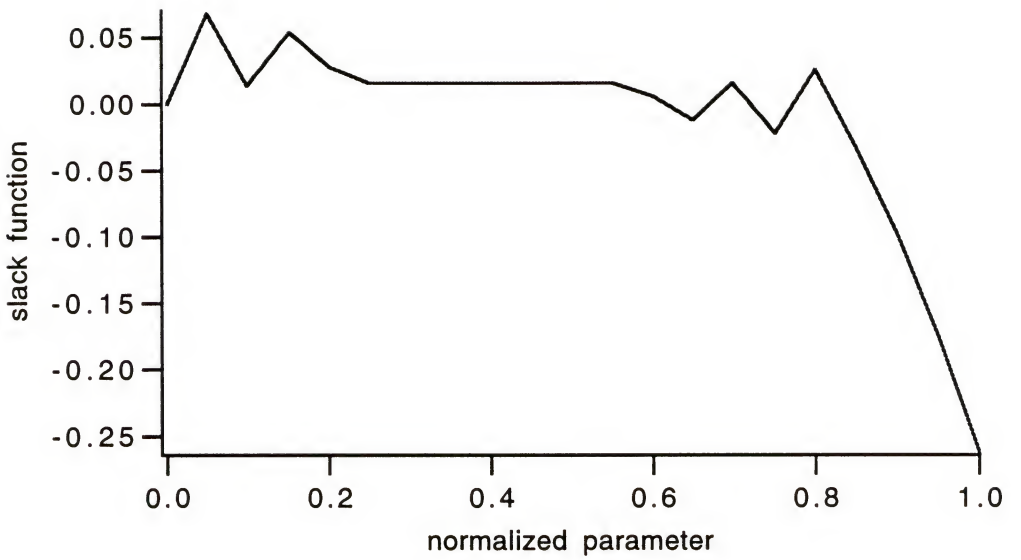


Figure 5.5 Slack Function for Path Planning around an Ellipsoid

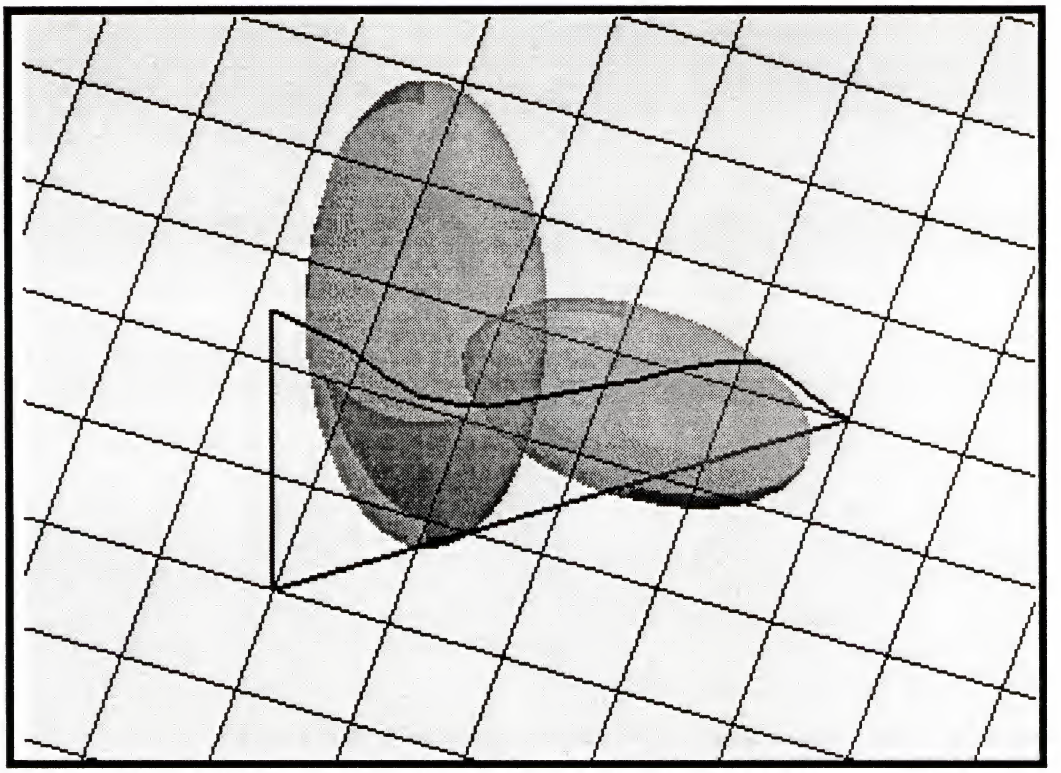


Figure 5.6 Path Planning Around Two Ellipsoids

Table 5.2 Algorithm Performance for Path Planning around Two Ellipsoids

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	20	140	Solution A	62	6.861	95.34

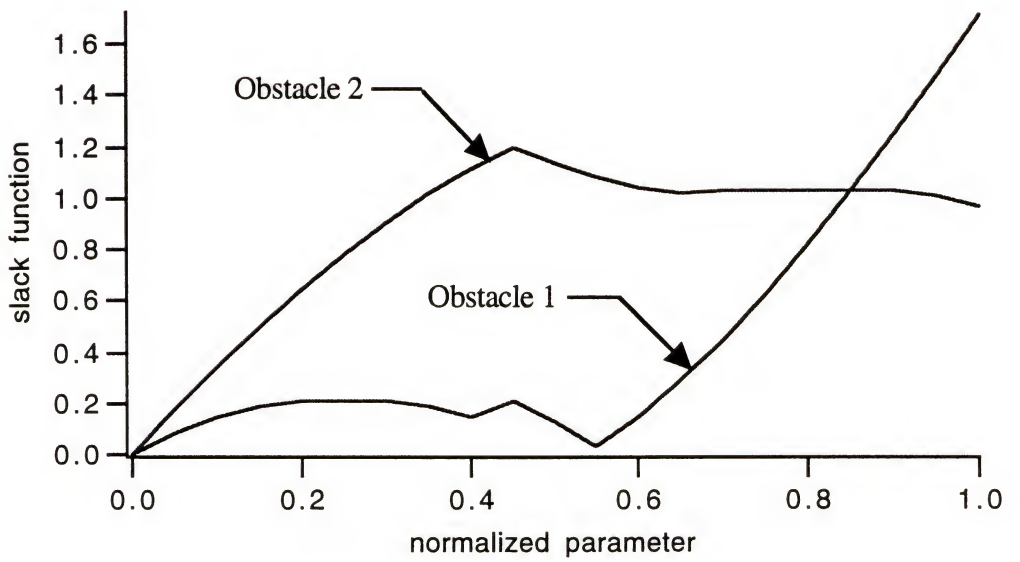


Figure 5.7 Slack Functions for Path Planning around Two Ellipsoids

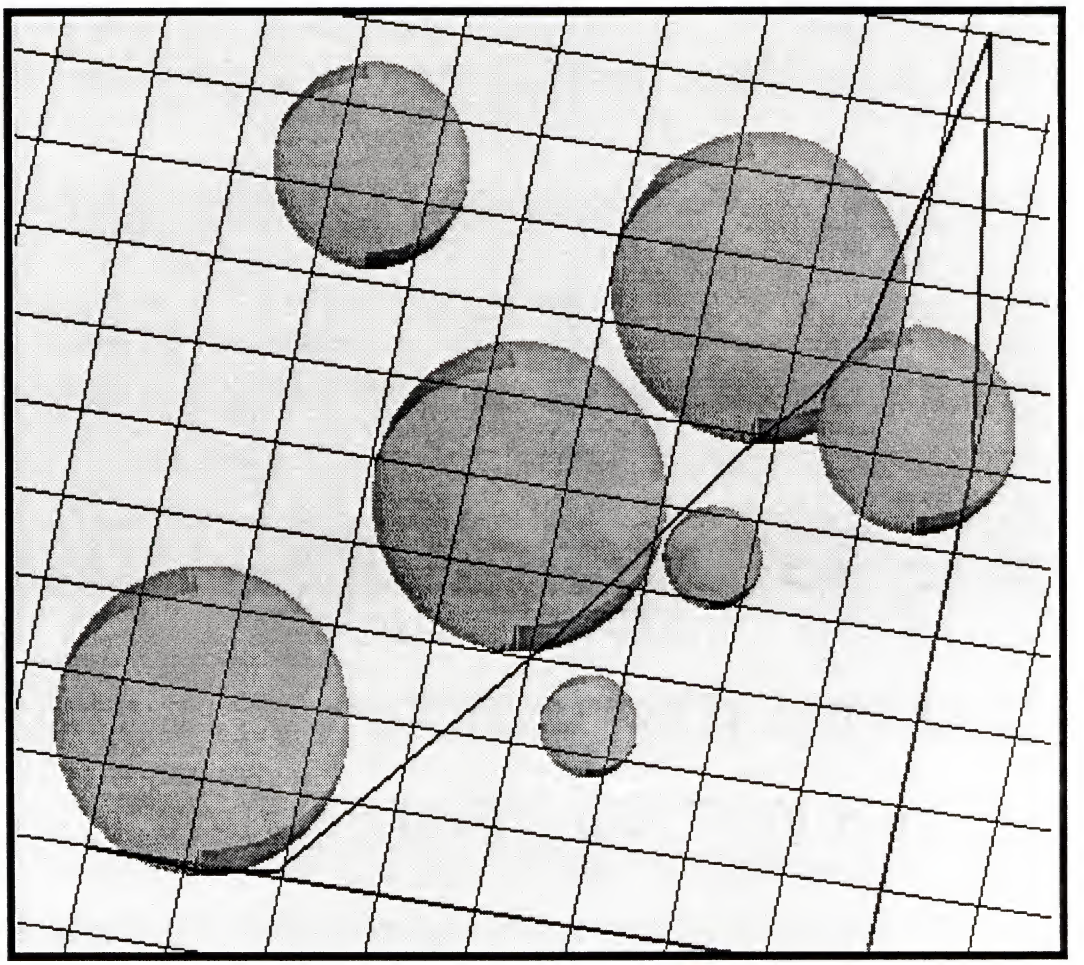


Figure 5.8 Path Planning around Spherical Obstacles

Table 5.3 Algorithm Performance for Path Planning around Spherical Obstacles

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	15	255	Solution A	49	13.319	190.29

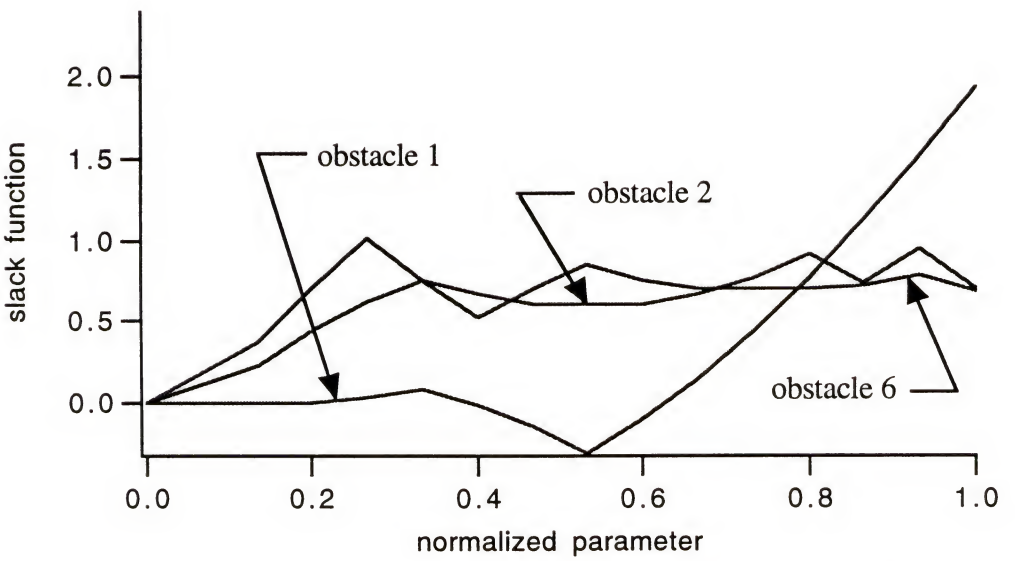


Figure 5.9 Slack Functions for Path Planning around General Spherical Obstacles

5.4 Stage Wise Optimization in Three Dimensional Space

In the next example a non convex composite obstacle is constructed from five intersecting spheres. This example is tested using both the standard and the stage wise algorithms. As can be seen by comparing Figures 5.10 and 5.12, the paths are not smooth but they do not violate the obstacle boundaries. A sharp contrast can be seen in the slack functions generated by the two algorithms. In Figure 5.11 the functions are jagged and irregular, whereas in Figure 5.13 they follow the idealized behavior as described in Chapter 3.

As a final example a composite body is constructed consisting of three ellipsoids, two of which have sharp curvatures. As can be seen in Figure 5.14 the problem is unstable and the standard algorithm generates an invalid path which violates the obstacle boundaries. When a solution to the problem was attempted using stage wise optimization the algorithm failed to converge leaving the problem unsolvable using the methods that have been developed.

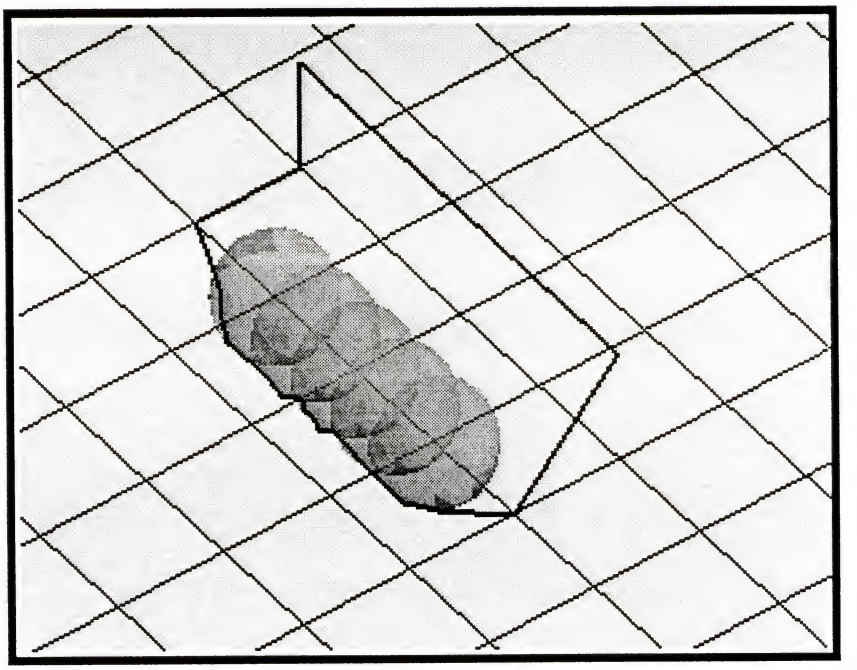


Figure 5.10 Path Planning around Five Intersecting Spheres

Table 5.4 Algorithm Performance for Path Planning around Spherical Obstacles

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	15	255	Solution A	49	13.319	190.29

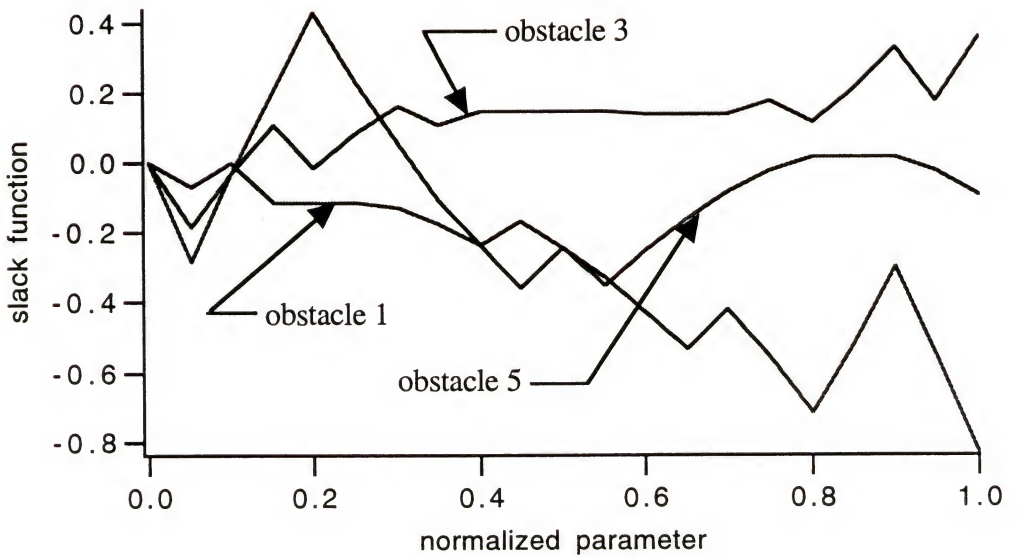


Figure 5.11 Slack Functions for Path Planning around Five Intersecting Spheres

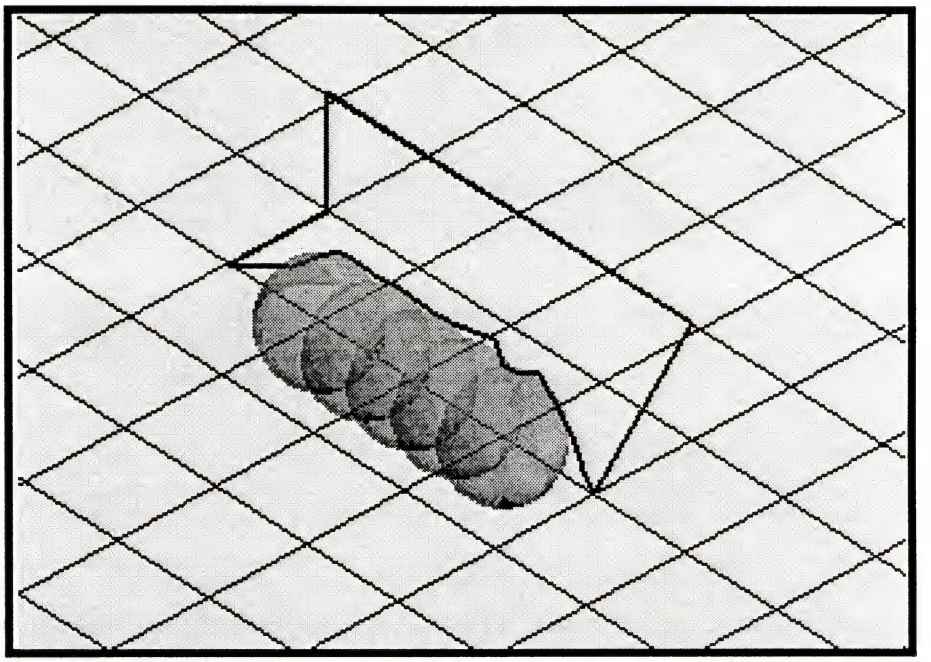


Figure 5.12 Stage Wise Optimization around Five Intersecting Spheres

Table 5.5 Algorithm Performance for Path Planning around Spherical Obstacles

Initial Guess	Number Steps	Size of Matrix	Convergence	Number of Iterations	Path Length	CPU time (Sec)
1	15	255	Solution A	49	13.319	190.29

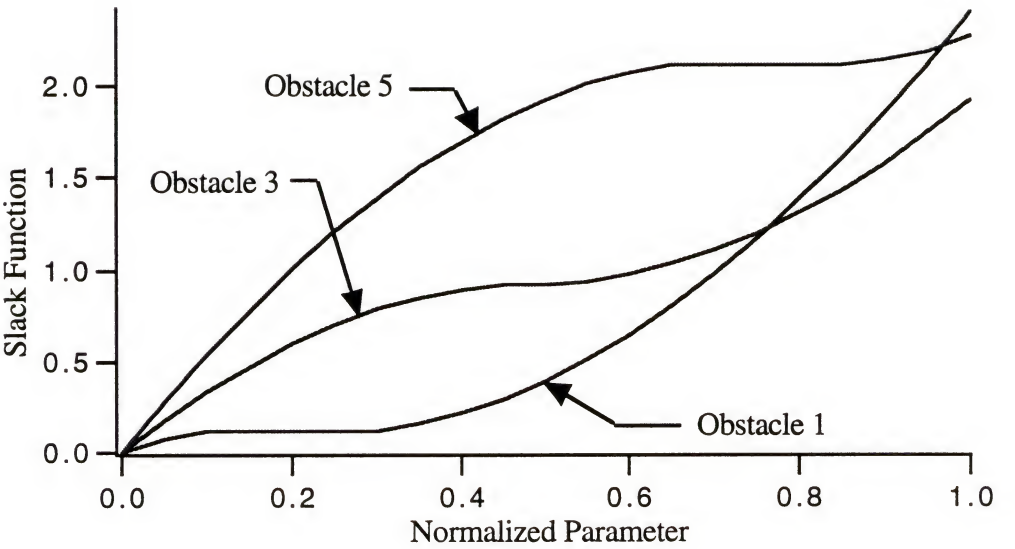


Figure 5.13 Slack Functions for the Stage Wise Optimization of Intersecting Spheres

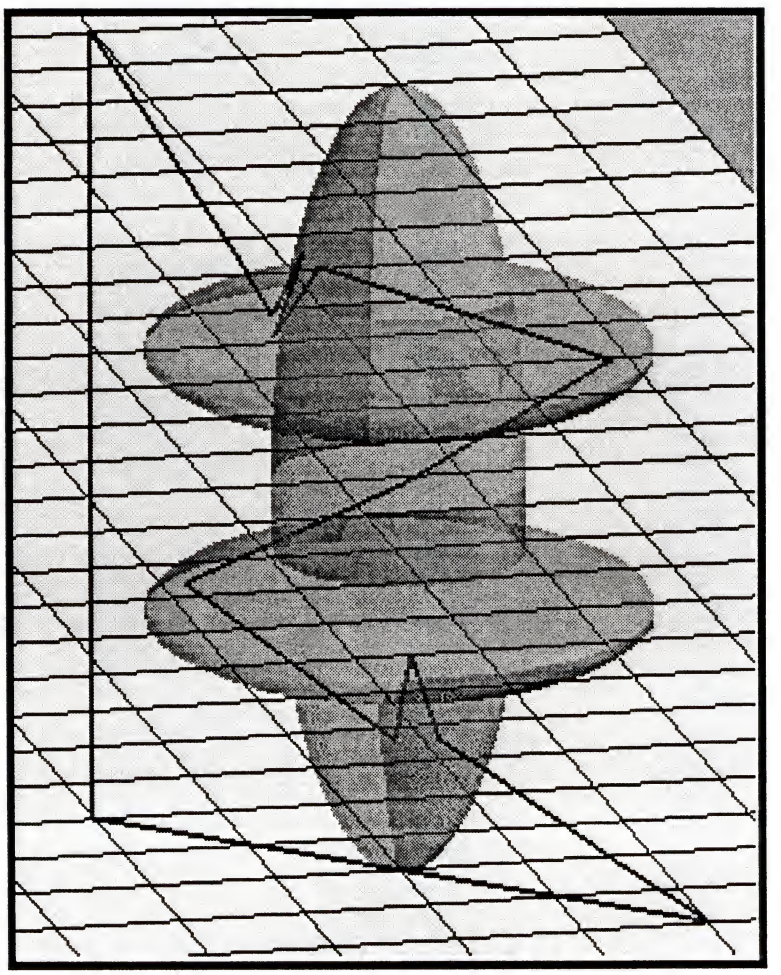


Figure 5.14 Failure of the Algorithm for a Composite Obstacle in Three Dimensions

CHAPTER 6
DISCUSSION AND CONCLUSIONS

6.1 Review of Path Planning in Two and Three Dimensional Space

At the outset of this dissertation it was explained that recent numerical techniques have been developed for solving calculus of variations problems, by Gregory and Lin [Greg 92], and that the goal of the research presented here was to explore the application of these techniques to path planning and obstacle avoidance. The derivation of the theoretical problem in Chapter 2 resulted in a problem formulation which is equivalent to the Problem of Bolza. As previously mentioned, the Problem of Bolza was first formulated to minimize the area of surfaces of revolution subject to constraints. In order to convert the theory into an algorithm in Chapter 3, the problem was transformed into one of finding the roots to a set of nonlinear finite difference equations which approximate the integral of the Euler-Lagrange equation. This algorithm has been tested for two- and three-dimensional problems in Chapters 4 and 5 respectively.

The performance of the algorithm has shown that it works well for simple geometries, but has difficulty generating valid paths as the complexity of the obstacle geometry increases. This type of observation can be applied to many numerical methods. For instance, it is well known in optimization theory that as the complexity of the objective function increases, algorithms have difficulty finding a minimum even though the problem is theoretically correct . A survey of these problems is given in Engineering Optimization, Methods and Applications [Rekl 83]. The errors caused by complexity in path generation fall into the following categories. The first dealing with slicing through the edges of the obstacles and the second dealing with the generation of invalid paths cause by parasitic roots. Also in some situations the solution diverges and becomes unstable.

The first of these problems can be explained by the nature of the approximation of the analytical solution. The theoretically exact solution curve solution is approximated by a finite set of points corresponding to the number of steps used in the approximation. Errors caused by slicing through objects can be generated in two ways. The first is due to the use of straight line segments approximating the solution curve in between two points. These segments in some instances will slice through the boundary of an obstacle whereas the theoretical curve would bend around the obstacle boundary. The second source of error is the approximation of these points itself. Gregory and Lin [Greg 92] have determined the maximum error between an approximated point and the actual point on the solution curve to be on the order of (h^2), where h is the step size used in the path.

The major mode of failure for the path planning algorithm is the generation of invalid paths which violate obstacle boundaries. This type of error appears as the complexity of the problem increases. It is felt that the source of this error is due to numerical instabilities in approximating the theoretical solution to the differential equations. A good discussion of instabilities involved in the numerical solution of differential equations is given in Numerical Methods [Horn 75] by Robert Hornbeck. In his discussion he describes the two types of solutions that can appear. The first is labeled as the fundamental solution and the second, of which there can be more than one, are the parasitic solutions. The fundamental solution is described as the solution that closely approximates the theoretical solution and will improve in its estimate as the step size approaches zero. This type of solution has been found in all the problems that have been tested. Parasitic solutions are so called because they feed on the numerical errors and "...the resulting solution no longer bears any resemblance to the exact solution." It is felt that these parasitic solutions cause the invalid paths in the algorithm. He further classifies the problems where the two types of solutions are present as numerically unstable and he proceeds to suggest changing to another numerical technique to solve the system of differential equations. Numerical instabilities can also create diverging and unbounded solutions. At the present time it is not clear what other techniques can be used to

numerically approximate the differential equations and whether or not new techniques will improve the overall performance of the algorithm. Intuitively it is felt that as the complexity of the problem increases a threshold will always be reached where instabilities appear. This is because the functions being approximated can always be made more complex by the addition of new obstacles. Unfortunately this threshold can be reached for a single obstacle in some circumstances.

From a practical point of view a qualitative test has been introduced, namely studying the form of the slack functions, which has been effective in categorizing valid or invalid paths in two dimensions. It seems fairly straightforward to extend this into a qualitative test which can then be developed into an algorithm. This could then be used as a check to see if a valid solution has been found.

6.2 Improvements to the Algorithm

The first task which must be addressed is development of a method for improving the algorithm so that the solution of the finite difference equations does not generate parasitic roots. At the present time it is not certain if this can be accomplished for the reasons given above. If this cannot be accomplished the scope of the algorithm is limited to small number of obstacles without sharp edges in Euclidean space. The only general method for solving systems of nonlinear differential equations is the use of finite difference equations with root finding.

The second set of improvements addresses issues of improving the speed of the algorithm. Most of the computational time is spent solving the system of matrix equations in each iterative step and this should be the major focus of any performance improvements. At present the algorithm uses an LU decomposition scheme for solving the system of equations. However the time required to reach a solution could be improved by taking advantage of the special characteristics of the matrix $[\partial G/\partial x]$. This matrix is block tri-diagonal which leads to improvements to the inversion algorithm. Gregory and Lin [Greg

92] also suggest that the matrix $[\partial G/\partial x]$ is approximately equivalent to $[\partial^2 f/\partial \dot{x}^2]$ which is itself positive definite. Whether or not $[\partial G/\partial x]$ is positive definite is not clear from their discussion. To determine positive definiteness for $[\partial G/\partial x]$ would require numerically calculating the eigenvalues of the matrix and checking to see if they are all positive. This is beyond the present scope of this research but may be another possible avenue of improving computational efficiency. A final possible improvement for the matrix inversion is to use iterative methods to solve the system of matrix equations. Due to the overall iterative nature of root finding using the Newton-Raphson method, a close initial guess could be obtained from the solution of the equations in the previous step. This initial guess could be used to seed the iterative method for finding a solution to the matrix equations in the current step. At the present time no indication can be given as to which algorithm will offer the best improvement but it is felt that they would have to be tried experimentally. All three of these algorithms are detailed in Matrix Computations [Golu 83] by Golub and Van Loan.

As can be seen from the exploratory studies, another possibility for improving the solution and speeding the search is by modifying the objective function. The examples considered in that chapter demonstrate the potential of such an avenue through the use of weighting factors on the energy and the slack functions. A final set of improvements to the algorithm concerns the modeling of more general geometries. This would involve approximation of the derivatives as opposed to analytically calculating them and conversion to a secant method. At the present time this increase in the scope of the problem should not be considered until the difficulties with instabilities have been resolved.

6.3 Difficulties Involved with Showing Sufficiency Conditions

One of the goals of this research was to determine sufficiency conditions for the path planning problem. In general sufficiency conditions for a calculus of variations problems can be determined. However, the task is much more complicated than the analogous situation in optimization theory. The candidate solutions determined from the

necessary conditions are first checked to make sure that they are imbeddable in a Mayer field. This is necessary to show that the candidate curve can be smoothly varied to obtain the curves in the neighborhood of the solution. This has no analogue in the sufficiency conditions for standard optimizations. A Hilbert integral and a Wierstrass excess function are then constructed from the candidate solution and checks are performed to determine optimality. This process is described in an Introduction to the Calculus of Variations [Sag 85] and is loosely analogous to the checking the second derivative of the Taylor series expansion of the function. The difficulty for the path planning problem is in the construction of the Mayer field. The usual method for constructing a field is to use conjugate point theory, but this requires that the candidate curve is regular or equivalently the matrix$[\partial^2 f/\partial \dot{x}^2]$ is nonsingular, where f is the function under the integral. For the Problem of Bolza this is not the case when the path comes in contact with an obstacle. For example given the problem of avoiding a single obstacle in the plane, the objective function is given by

$$I = \int_0^1 (\dot{x}_1^2 + \dot{x}_2^2 - \dot{x}_3\{(x_1-a)^2 + (x_2-b)^2 - r^2 - \dot{x}_4^2\})\, dt \qquad (6.1).$$

The matrix $[\partial^2 f/\partial \dot{x}^2]$ is given by,

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2\dot{x}_4 \\ 0 & 0 & 2\dot{x}_4 & 2\dot{x}_3 \end{bmatrix} \qquad (6.2).$$

Clearly the matrix becomes singular when the path comes in contact with an obstacle and the slack derivatives $\dot{x}_4$ become zero. The Mayer field then has to be specially constructed for the particular problem under consideration to account for these points. This has been done for several cases where PhD mathematics students, as part of their research, constructed the criteria for sufficiency conditions in particular applications of the Problem of Bolza. These cases are detailed in Contributions to the Calculus of Variations [Vale 37].

6.4 Extension of the Problem to Include Other Metrics

A final goal of this research was to allow for the use of other metrics to describe and quantify more complicated problems such as rigid body motion. One of the advantages of the method that has been developed is that it remains very close to the underlying mathematics. Once a metric has been proposed for comparing path lengths in a more general sense, it is straightforward to apply this to path planning using the theory that has been developed. Whether or not the numerical difficulties can be resolved is another matter. The long range advantage of this approach is that it could be applied to in many different types of problems with a minimum of modification.

REFERENCES

[Ahri 94] Ahrikencheikh, C., and Seireg, A., 1994, *Optimized-Motion Planning*, John Wiley and Sons, Inc., New York, NY.

[Arno 83] Arnol'd, V.I., 1984, "Singularities in Variational Calculus", Journal of Soviet Mathematics, Vol.27, pp. 2679 - 2713.

[Cela 92] Cela, A., and Haman, Y., 1992, "Optimal Motion Planning of a Multiple-Robot System Based on Decomposition Coordination," IEEE Transactions on Robotics and Automation, Vol. 8, No. 5, pp. 585 - 596.

[Gilb 85] Gilbert, E., and Johnson, W., 1985, "Distance Functions and Their Application to Robot Path Planning in the Presence of Obstacles," IEEE Journal of Robotics and Automation, Vol. RA-1, No. 1, pp. 21 - 30.

[Gilb 88] Gilbert, E., Johnson, W., and Keerthi, S., 1988, "A Fast Procedure for Computing the Distance Between Complex Objects in Three-Dimensional Space," IEEE Journal of Robotics and Automation, Vol. 4, No. 2, pp. 193 - 203.

[Give 90] Givental, A. B., 1990, "Singular Lagrangian Manifolds and Their Lagrangian Maps," Journal of Soviet Mathematics, Vol. 52, No. 4, pp. 3246 - 3277.

[Golu 83] Golub, G. H., and Van Loan, C.F., 1983, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD.

[Greg 92] Gregory, J., and Cantian, L., 1992, *Constrained Optimization in the Calculus of Variations and Optimal Control Theory*, Van Nostrand Reinhold, New York, NY.

[Hart 68] Hart, P.E., Nilsson, N.J., and Raphael, B., 1968, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," IEEE Transactions on Systems, Science and Cybernetics, Vol SSC-4, No. 2, pp. 100 - 107.

[Horn 75] Hornbeck, R. W., 1975, *Numerical Methods,* Prentice-Hall Inc., Englewood Cliffs, NJ.

[Hu 93] Hu, T., Kahng, A., and Robins, G., 1993, " Optimal Robust Path Planning in General Environments," IEEE Journal of Robotics and Automation, Vol. 9, No 6, pp. 775 - 784.

[Jone 87] Jones, A., Gray, A., and Hutton, Robert, 1987,*Manifolds and Mechanics*, ed. Morris, S., Cambridge University Press, Cambridge England.

[Khat 86] Khatib, O., 1986," Real Time Obstacle Avoidance for Manipulators and Mobile Robots," International Journal of Robotics Research, Vol. 5, No. 1, pp. 90 - 98.

[Lato 91] Latombe, J., 1991, *Robot Motion Planning,* Kluwer Academic Publishers, Norwell, MA.

[Love 75] Lovelock, D., and Rund, H., 1975, *Tensors, Differential Forms, and Variational Principles*, Dover Publications Inc., New York, NY.

[Loza 83] Lozano-Perez, T., 1983, "Spatial Planning: A Configuration Approach," IEEE Transactions on Computers, Vol. C-32, No. 2, pp. 108-120.

[Luh 81] Luh, J., and Lin, C., 1981 "Optimum Path Planning for Mechanical Manipulators," Journal of Dynamic Systems Measurement and Control, Vol. 102, pp.142 - 151.

[Lum1 85] Lumelsky, V., 1985, "On Dynamic Path Planning for a Planar Robot Arm", Report No. 8505, Dept. of Electrical Engineering, Yale University, New Haven, CT.

[Lum2 85] Lumelsky, V., 1985, "Continuous Path Planning for Various Plannar Robot Arm Configurations", Report No. 8506, Dept. of Electrical Engineering, Yale University, New Haven, CT.

[Lum3 85] Lumelsky, V., 1985, "Continuous Path Planning for a Three-Dimensional Cartesian Arm", Report No. 8507, Dept. of Electrical Engineering, Yale University, New Haven, CT.

[Orte 70] Ortega, J.M., and Rheinboldt W.C., 1970 *Iterative Solution of Nonlinear Equations in Several Variables,* Academic Press, New York and London.

[Pres 88] Press, W., Flannery, B., Teukolsky, S., and Vetterling, W., 1988, *Numerical Recipes in C, The Art of Scientific Computing*, Cambridge University Press, Cambridge England.

[Rekl 83] Reklaitis, G.V., Ravindran, A., Ragsdell, K.M., 1983, *Engineering Optimization, Methods and Applications*, John Wiley and Sons, New York, N.Y.

[Saga 85] Sagan, H., 1985, *Introduction to the Calculus of Variations*, Dover Publications Inc., New York, NY.

[Shch 88] Shcherbak, O.P., 1988, "Wavefronts and Reflection Groups", Russian Mathematical Surveys, Vol. 43, No. 3, pp. 149 - 194.

[Shin 86] Shin, K., and McKay, N., 1986, "Selection of Near Minimum Time Geometric Paths for Robotic Manipulators," IEEE Transactions on Automatic Control, Vol. AC-31, No. 6, pp. 501 - 511.

[Shin 85] Shin, K., and McKay, N., 1985, " A Dynamic Programming Approach to Trajectory Planning of Robotic Manipulators," IEEE Transactions on Automatic Control, Vol. AC-31, No. 6, pp. 491 - 500.

[Spiv 79] Spivak, M., 1979, *A Comprehensive Introduction to Differential Geometry*, 2 ed., Publish or Perish, Inc. Berkeley, CA.

[Sun 92] Sun, K. and Lumelsky, V., 1992 " Path Planning Among Unknown Obstacles: The Case of a Three-Dimensional Cartesian Arm," IEEE Transactions on Robotics and Automation, Vol. 8, No. 6, pp. 776 - 786.

[Vale 37] Valentine, F.A. et al, 1933-1937, *Contributions to the Calculus of Variations*, University of Chicago Press, Chicago IL.

John Damian Duffy was educated at Buchholz High School in Gainesville Florida. He then went on to the University of Florida where he received a Bachelor of Science in mechanical engineering with honors in December 1988. He traveled to Troy, New York, to attend Rensselaer Polytechnic Institute where he received a Master of Science in mechanical engineering in December 1990. He returned to the University of Florida in January 1992 to commence work on a Doctorate of Philosophy degree in mechanical engineering.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Ali A. Seireg, Chairman
Ebaugh Professor of
Mechanical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Ralph G. Selfridge
Professor of Computer and
Information Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Gary K. Matthew
Associate Professor of
Mechanical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

K. Scott Smith
Associate Professor of
Mechanical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Allen Lush
Assistant Professor of
Mechanical Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy,

December, 1994

Winfred M. Phillips
Dean, College of Engineering

Karen A. Holbrook
Dean, Graduate School